Mechatronics 23 (2013) 677–688

Contents lists available at ScienceDirect

Mechatronics

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

A bumpless hybrid supervisory control algorithm for the formation of unmanned helicopters ☆

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

[a] The Department of Electrical and Computer Engineering, North Carolina A&T State University, Greensboro, NC 27411 USA
[b] Department of Electrical Engineering, University of Notre Dame, Notre Dame, USA
[c] Graduate School for Integrative Sciences and Engineering (NGS), and the Department of Electrical and Computer Engineering (ECE), National University of Singapore, Singapore

ARTICLE INFO

Article history:
Received 12 October 2012
Accepted 4 July 2013
Available online 22 August 2013

Keywords:
Formation control
Hybrid supervisory control
Unmanned Aerial Vehicles (UAVs)
Implementation issues

ABSTRACT

This paper presents a bumpless hybrid supervisory control scheme for the formation of unmanned helicopters. The approach is based on the polar partitioning of the space, from which a finite bisimilar quotient transition system of the original continuous variable control system is obtained. Then, to implement the designed hybrid supervisory control algorithm, a hierarchical control structure is introduced with a discrete supervisor on the top layer that is connected to the regulation layer via an interface layer. Transiting over the partitioned space may cause jumps on the generated control signal which is harmful for a real flight system. Hence, a smooth control mechanism is introduced that has no jump when the system's trajectory transits from one region to its adjacent region while preserving the bisimulation relation between the abstract model and the original partitioned system. Several actual flight tests have been conducted to verify the algorithm and the control performance.

1. Introduction

Formation of the Unmanned Aerial Vehicles (UAVs) can leverage the capabilities of the team to have more effective performance in missions such as cooperative SLAM, coverage and recognisance, and security patrol [1–3]. Hence, recent years have seen an increasing interest in the study of UAV formation control from both theoretical and experimental points of view. In the literature there are some methods that can partly address the formation problem. For example, in [4–6], the problem of *reaching the formation* is investigated using optimal control techniques, navigation function, and potential field approaches. *Keeping 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 [7–10]. Finally, in [11–13], different mechanisms for *collision avoidance* have been introduced using probabilistic methods, MILP programming, and behavioral control. Most of these methods are suitable just for certain aspects of these formation tasks. The traditional practice is to design controllers for each task separately and switch between them based on different situations. However, the separate design of switching logic and continuous controllers is problematic as unexpected behaviors could be generated due to switching between the sub-controllers. This calls for a unified way to design formation controller and switching logic. In our recent study [14], a unified framework was introduced to address all aspects of a formation control mission based on hybrid control theory [15] which can integrate the analysis and design of both the discrete-event dynamics and the continuous evolution of the systems. In particular, the approach introduced in [14] was rooted from hybrid supervisory control [15]. The basic idea is to use polar abstraction of the motion space and utilize the properties of multi-affine functions [16] over the partitioned space. The abstraction technique [17] can convert the original continuous system with infinite states into a finite state machine for which one can use the well developed theory of supervisory control of discrete event systems (DES) [18]. Subjected to the bisimulation relation between the abstracted system and the original continuous system, their behavior will be the same so that the discrete supervisor, designed for the discrete finite model, can be applied to the original system.

Here, the key is how to implement this hybrid controller. For this purpose, we introduce a hierarchical hybrid supervisory control structure which has a discrete supervisor on the top and a continuous low level control on the low layer. To connect the discrete supervisor to the continuous low level, an interface layer is introduced which on the one hand interprets the continuous signals for the discrete supervisor and on the other hand, converts the generated discrete symbols to continuous control signals to be applied to the low layer. Based on the decision made by the supervisor, the

☆ Financial supports from NSF-CNS-1239222 and NSF-EECS-1253488for this work are greatly acknowledged. A primary version of in this paper was submitted for presentation in the 2013 American Control Conference.
* Corresponding author. Tel.: +1 574 6313177.
E-mail address: hlin1@nd.edu (H. Lin).

0957-4158/$ - see front matter © 2013 Elsevier Ltd. All rights reserved.
http://dx.doi.org/10.1016/j.mechatronics.2013.07.004

discrete commands would change when the system's trajectory passes from one region to another region in the partitioned space. A very important problem here is that the generated control signal may have jumps when the system transits from one region to another one. These kind of jumps in the generated control signal may cause serious problems for a real flight system. Therefore, here we propose an algorithm which can generate a smooth control signal applicable to the low level continuous layer. The basic idea is to tune the value of the vector field at the vertices of the partitioning elements at the common edges to provide a smooth control signal while preserving the bisimilarity relation between the abstracted model and the original continuous system.

Hence, this paper presents a smooth hybrid supervisory control algorithm for the formation of UAV helicopters and focuses on the implementation issues of the proposed algorithm. More specifically, our main contributions in this paper are that firstly, an interface layer is introduced to connect the discrete supervisor layer to the continuous plant. This interface layer is responsible for converting the continuous signals of the plant into some symbols understandable by the discrete supervisor, and vice versa. Secondly, the time scheduling of the events being generated by the system has been investigated and has been correspondingly considered in the implementation of the supervisor. Thirdly, a control scheme is proposed to smoothly transit through the partitioning elements so that there is no jump in the generated control signal when the system transits from one region to its adjacent regions. Finally, a cooperative testbed is developed and the proposed algorithm has been verified through actual flight tests.

The rest of this paper is organized as follows. First, the developed cooperative testbed is explained in Section 2. Then, Section 3 describes the preliminaries of the hybrid supervisory control algorithm for a formation mission. In Section 4, a hierarchical hybrid control structure is proposed which has a discrete supervision layer on the top that is connected to the continuous low layer via an interface layer. Section 5 describes implementation issues for the algorithm and provides a mechanism to generate a smooth control signal. Flight test results are demonstrated in Section 6. The paper is concluded in Section 7.

2. Test-bed infrastructure

For the implementation of the proposed hybrid formation algorithm we have used a set of two UAV helicopters, HeLion and SheLion (Fig. 1) which are developed by our research group at the National University of Singapore.

These UAVs are radio-controlled helicopter, Raptor 90. The size of these helicopters is 1410 mm in length and 190 mm in width of the fuselage. The maximum takingoff weight is 11 kg including 5 kg as the dry weight of helicopter and 6 kg as the effective payload. Their main rotors and tail rotors have the diameter of 1605 mm and 260 mm, respectively.

Fig. 1. NUS cooperative UAVs test-bed.

These helicopters have been provided with an avionic system that make them able to autonomously accomplish different individual or cooperative maneuvers. Their avionic systems are equipped with a PC/104 ATHENA, as an onboard airborne computer system which has four RS-232 serial ports, a 16-pin digital to analog (D/A) port, two counters/timers and runs at 600 MHz.

Moreover, for the navigation a compact fully integrated INS/GPS, NAV 420, Crossbow, is used to provide three-axis velocities, acceleration, and angular rates in the body frame, as well as longitude, latitude, relative height, and heading, pitch, and roll angles. For the reliable communication between the UAVs, and also between the UAVs and the ground station, we have used serial wireless radio modems, IM-500X008, FreeWave, with the working frequency of 2.4 GHz, which can cover a wide range up to 32 km in an open field environment.

The onboard program is implemented using QNX Neutrino real time operating system. For this onboard program a multi-thread structure is developed which includes several threads for flight control; reading from data acquisition board; driving the servo actuators; making dual-directional wireless communication with other UAVs or with the ground station; and logging data in an onboard compact flash card.

Furthermore, for these helicopters, a hardware-in-the-loop simulation software has been developed by integrating the developed hardware and embedded software together with the nonlinear dynamic model of the UAV helicopters. 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. Consequently, the simulation results of this simulator are very close to the actual flight tests, and it can provide a safe and reliable environment for the pre-evaluation of the control algorithms.

The modeling and low level control structure of the NUS UAV helicopters are explained in [19–21]. For the regulation layer of

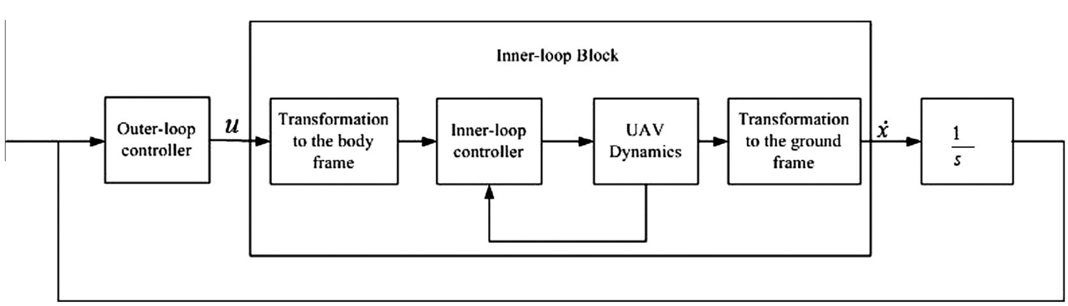

Fig. 2. The control structure of the NUS UAVs.

these helicopters we have proposed a two-layer control structure in which the inner-loop controller stabilizes the system using H_∞ control design techniques, and their outer-loop is used to derive the system towards the desired location (Fig. 2). As it has been discussed in [20], in this control structure, the inner-loop is fast enough to track the given references, so that the outer-loop dynamics can be approximately described as follows:

$$\dot{x} = u, \quad x \in \mathbb{R}^2, \quad u \in U \subseteq \mathbb{R}^2, \tag{1}$$

where x is the position of the UAV; u is the UAV velocity reference generated by the formation algorithm, and U is the convex set of velocity constraints.

3. Preliminaries on hybrid formation control

In a leader follower formation scenario, consider the follower velocity in the following form:

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

For these helicopters, our aim is to design the formation controller to generate the relative velocity of the follower, V_{rel}, such that starting from any initial point inside the control horizon, it eventually reaches the desired relative distance with respect to the leader, while avoiding the collision between the leader and the follower. Moreover, after reaching the formation, the follower UAV should remain at the desired position with respect to the leader.

To solve this problem, in [14], a method is introduced for the polar abstraction of the motion space which uses the properties of multi-affine vector fields over the polar partitioned space. Within this framework, a DES model can be achieved for which we can design a decentralized supervisor to achieve three major goals: *reaching the formation*, *keeping the formation*, and *collision avoidance*. This method is briefly explained in the following sections.

3.1. Polar partitioning of the state space

Consider a relatively fixed frame, in which the follower moves with the velocity of V_{rel} and the leader has a relatively fixed position. In this framework, imagine a circle with the radius of R_m that is centered at the desired position of the follower. With the aid of the partitioning curves $\{r_i = \frac{R_m}{n_r-1}(i-1),\ i = 1,\dots,n_r\}$ and $\{\theta_j = \frac{2\pi}{n_\theta-1}(j-1),\ j = 1,\dots,n_\theta\}$, this circle can be partitioned into $(n_r-1)(n_\theta-1)$ partitioning elements. An element $R_{i,j} = \{p = (r,\theta) | r_i \leqslant r \leqslant r_{i+1},\ \theta_j \leqslant \theta \leqslant \theta_{j+1}\}$, has four vertices, v_0, v_1, v_2, v_3 (Fig. 3(a)), four edges, E_r^+, E_r^-, E_θ^+, E_θ^- (Fig. 3(b)), and correspondingly, four outer normal vectors n_r^+, n_r^-, n_θ^+, n_θ^- (Fig. 3(c)). In region $R_{i,j}$, the notation $E_{p,q}$ is used for the edge which is incident with the vertices v_p and v_q, and correspondingly, $n_{p,q}$ is used to denote its outer normal vector.

To implement the formation algorithm, we will deploy multi-affine functions over the partitioned space. A multi-affine function $f: \mathbb{R}^n \to \mathbb{R}^m$, has the property that for any $1 \leqslant i \leqslant n$ and any $a_1,\ a_2 \geqslant 0$ with $a_1 + a_2 = 1$, $f(x_1,\dots,(a_1 x_{i_1} + a_2 x_{i_2}),\ x_{i+1},\dots x_n) = a_1 f(x_1,\dots,x_{i_1},x_{i+1},\dots x_n) + a_2 f(x_1,\dots,x_{i_2},x_{i+1},\dots x_n)$. The following proposition shows that the value of a multi-affine function over the partitioning element $R_{i,j}$, can be uniquely expressed in terms of the values of the function at the vertices of $R_{i,j}$.

Proposition 1 [14] *Consider a multi-affine function $g(x): \mathbb{R}^2 \to \mathbb{R}^2$ over the region $R_{i,j}$. The following property always holds true:*

$$\forall x = (r,\theta) \in R_{i,j} : g(x) = \sum_{m=0}^{3} \lambda_m g(v_m), \tag{3}$$

where λ_m, $m = 0, \dots, 3$, are obtained as follows:

$$\lambda_m = \lambda_r^{\Psi_r(v_m)}(1-\lambda_r)^{1-\Psi_r(v_m)}\lambda_\theta^{\Psi_\theta(v_m)}(1-\lambda_\theta)^{1-\Psi_\theta(v_m)}, \tag{4}$$

where $\lambda_r = \frac{r-r_i}{r_{i+1}-r_i}$, $\lambda_\theta = \frac{\theta-\theta_j}{\theta_{j+1}-\theta_j}$, $\Psi_r(v_m) = \begin{cases} 0 & m = 0,2 \\ 1 & m = 1,3 \end{cases}$ *and* $\Psi_\theta(v_m) = \begin{cases} 0 & m = 0,1 \\ 1 & m = 2,3 \end{cases}$.

Remark 1. It can be verified that the resulting coefficients λ_m, $m = 0, 1, 2, 3$, have the property that $\lambda_m \geqslant 0$ and $\sum_m \lambda_m = 1$.

The above proposition holds true for the edges as described in the following corollary.

Corollary 1. *For a multi-affine function g(x) defined over the element $R_{i,j}$ and for all of the edges E_q^s of $R_{i,j}$, $q \in \{r,\theta\}$ and $s \in \{+,-\}$, the following property holds true:*

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

where λ_m can be obtained as follows:

- *For edges E_r^+ and E_r^- : $\lambda_m = \lambda_\theta^{\Psi_\theta(t)}(1-\lambda_\theta)^{1-\Psi_\theta(t)}$.*
- *For edges E_θ^+ and E_θ^- : $\lambda_m = \lambda_r^{\Psi_r(u)}(1-\lambda_r)^{1-\Psi_r(u)}$.*

Using these properties of multi-affine functions, it is possible to flexibly design a hierarchical control structure for the formation control of the UAVs as described in the following section.

4. Hierarchical control structure for the formation of unmanned helicopters

For the above discussed model of the plant defined over the partitioned space, we will design a discrete supervisor which pushes the system trajectories to pass through the desired regions to achieve the desired behavior. The designed discrete supervisor cannot be directly connected to the continuous plant. Hence, it is required to construct an interface layer which can translate continuous signals of the plant to a sequence of discrete symbols understandable for the supervisor. Also, the interface layer is responsible for converting discrete commands received from the supervisor, to continuous control inputs to be given to the plant. These two jobs are respectively realized by the blocks Detector and Actuator embedded in the interface layer as it is shown in Fig. 4. The elements of this control hierarchy are discussed in the following parts.

4.1. The interface layer

4.1.1. The detector block

When the system's trajectory crosses the boundaries of the region, a detection event will be generated which informs the supervisor that the system has entered a new region.

More specifically, a detection event $d_{i,j}$ will happen at $t(d_{i,j})$ when the system's trajectory $x(t)$ satisfies the following conditions:

- $\exists \tau > 0$ such that $x(t) \notin R_{i,j}$ for $t \in (t(d_{i,j}) - \tau,\ t(d_{i,j}))$.
- $\exists \tau_d > 0$ such that $x(t) \in R_{i,j}$ for $t \in [t(d_{i,j}),\ t(d_{i,j}) + \tau_d)$.

Also, if the leader position is on the way of the follower towards the desired position, the event *Ob* will be generated to inform the supervisor about the risk of collision.

4.1.2. The actuator block

Having the information about the newly entered region, the supervisor can issue a discrete command to push the system trajectory to move towards the desired region. However, the discrete symbols generated by the supervisor need to be translated to a continuous form. For this purpose, the properties of multi-affine functions are utilized by which we can design continuous controllers that drive the system's trajectory to either stay in the current region for ever (invariant region) or exit from one of its edges (exit edge). Next, the invariant region and the exit edge are formally defined and the sufficient conditions which make a region invariant or one of its edges an exit edge are investigated.

Definition 1 (*Invariant region*). In the circle C_{R_m} and the vector field $\dot{x} = g(x),\ g : \mathbb{R}^2 \to \mathbb{R}^2$, the region $R_{i,j}$ is said to be invariant region, if $\forall x(0) \in int(R_{i,j})$, and $x(t) \in R_{i,j}$ for $t \geqslant 0$.

The following theorem and corollary show how we can construct an invariant region:

Theorem 1. *Given a continuous multi-affine vector field $\dot{x} = g(x),\ g : \mathbb{R}^2 \to \mathbb{R}^2$, defined over the region $R_{i,j}$, the systems trajectory cannot leave the region through the edge $E_{p,q}$ with the outer normal $n_{p,q}$ if $n_{p,q}(y)^T g(v_m) < 0$, for all $v_m \in \{v_p, v_q\}$ and all $y \in E_{p,q}$.*

Proof. According to Corollary 1, $\forall x \in E_{p,q} : g(x) = \sum_{v_m} \lambda_m g(v_m)$, $v_m \in \{v_p, v_q\}$. Substituting this value of $g(x)$, we will have $n_{p,q}(y)^T g(x) = n_{p,q}(y)^T.\ \sum_{v_m} \lambda_m g(v_m) = \sum_{v_m} \lambda_m\, n_{p,q}(y)^T \cdot g(v_m)$. Since, $n_{p,q}(y)^T g(v_m) < 0$ for both $v_m = v_p$ and $v_m = v_q$, and all $y \in E_{p,q}$, and since $\lambda_m \geqslant 0$ and $\sum_{m \in \{p,q\}} \lambda_m = 1$, it can be concluded that $n_{p,q}(y)^T.g(x) < 0$ for all x, $y \in E_{p,q}$, which means that the trajectories of the system cannot leave $R_{i,j}$ through the edge $E_{p,q}$. □

Corollary 2 (*Sufficient condition for $R_{i,j}$ to be an invariant region*). *For a continuous multi-affine vector field $\dot{x} = h(x, u(x)) = g(x),\ h : \mathbb{R}^2 \to \mathbb{R}^2,\ R_{i,j}$ is an invariant region if there exists a controller $u : \mathbb{R}^2 \to U \subseteq \mathbb{R}^2$, such that for each vertex v_m, m = 0, 1, 2, 3, with incident edges $E_q^s \in E(v_m)$, and corresponding outer normals $n_q^s,\ q \in \{r, \theta\}$ and $s \in \{+, -\}$:*

$$U_m = U \cap \Big\{ u \in \mathbb{R}^2 | n_q^s(y)^T \cdot g(v_m) < 0,\ \text{for all } E_q^s \in E(v_m),\ \text{and for all } y \in E_q^s \Big\} \neq \emptyset, \tag{6}$$

where the convex set U represents the velocity bounds.

Proof. If (6) holds true, since $U_m \neq \emptyset$, there exists $u_m \in U_m$, m = 0, 1, 2, 3, such that based on Theorem 2, the value of the vector field at the vertices does not let the trajectory of the system leave the region from any of the edges. □

The exit edge then can be defined as follows:

Definition 2 (*Exit edge*). In the circle C_{R_m} and the vector field $\dot{x} = g(x),\ g : \mathbb{R}^2 \to \mathbb{R}^2$, the edge $E_q^s,\ q \in \{r, \theta\}$ and $s \in \{+, -\}$, is said to be an exit edge, if $\forall x(0) \in int(R_{i,j})$, there exist τ(*finite*) > 0 and $\tau_d > 0$ satisfying:

1. $x(t) \in int(R_{i,j})$ for $t \in [0, \tau)$,
2. $x(t) \in E_q^s$ for $t = \tau$,
3. $x(t) \notin R_{i,j}$ for $t \in (\tau, \tau + \tau_d)$.

The following theorem shows the way that we can construct an exit edge:

Theorem 2 (*Sufficient condition for an exit edge*). *For a continuous multi-affine vector field $\dot{x} = h(x, u(x)) = g(x),\ g : \mathbb{R}^2 \to \mathbb{R}^2$, the edge E_q^s with the outer normal n_q^s, $q \in \{r, \theta\}$ and $s \in \{+, -\}$, is an exit edge if there exists a controller $u : \mathbb{R}^2 \to U \subseteq \mathbb{R}^2$, such that for each vertex v_m, m = 0, 1, 2, 3, the following property holds true:*

$$\begin{aligned} U_m = U \cap \Big\{ & u \in \mathbb{R}^2 |\ n_q^s(y)^T \cdot g(v_m) > 0,\ \text{for all } v_m \text{ and all } y \in E_q^s \Big\} \cap \\ & \Big\{ u \in \mathbb{R}^2 | n_{q'}^{s'}(y)^T \cdot g(v_m) < 0,\ \text{for all } v_m \in V\big(E_{q'}^{s'}\big) \text{ with} \\ & E_{q'}^{s'} \neq E_q^s \text{ and all } y \in E_{q'}^{s'} \Big\} \neq \emptyset, \end{aligned} \tag{7}$$

where the convex set U represents the velocity bounds.

Proof. If $U_m \neq \emptyset$, there exists $u_m \in U_m$, such that $n_{q'}^{s'}(y)^T \cdot g(v_m) < 0$, for all $E_{q'}^{s'} \neq E_q^s$ and all $y \in E_{q'}^{s'}$. Therefore, based on Theorem 1, the trajectories of the system do not leave $R_{i,j}$ through the non-exit edges. On the other hand, we have $n_q^s(y)^T \cdot g(v_m) > 0$ for all v_m and all $y \in E_q^s$. According to Proposition 1, for the multi-affine function g, there exist λ_m such that $\forall x \in R_{i,j} : g(x) = \sum_m \lambda_m g(v_m),\ m = 0,\ 1,\ 2,\ 3.$ Since $\lambda_m \geqslant 0$ and $\sum_m \lambda_m = 1$, then $n_q^s(y)^T \cdot \lambda_m g(v_m) \geqslant 0$ for all v_m and all $y \in E_q^s$. This will lead to have $n_q^s(y)^T \cdot g(x) > 0$ for all $x \in \overline{R}_{i,j}$, which means that the trajectories of the system have a strictly positive velocity in the direction of n_q^s steering them to leave $R_{i,j}$ through the edge E_q^s. □

Solving the inequalities given in Theorem 2 and Corollary 2, for the system dynamics given in (1), the following control values at the vertices of the region $R_{i,j}$ can make it an invariant region or can make one of its edges an exit edge. For the invariant controller, the control label is C_0 and the control values at the vertices are:

$$\begin{cases} u(v_0) = 1\angle\big(\theta_j + 0.5\big|\theta_j - \theta_{j+1} + \frac{\pi}{2}\big|\big) \\ u(v_1) = 1\angle\big(\theta_j + \pi - 0.5\big|\theta_j - \theta_{j+1} + \frac{\pi}{2}\big|\big) \\ u(v_2) = 1\angle\big(\theta_{j+1} - 0.5\big|\theta_j - \theta_{j+1} + \frac{\pi}{2}\big|\big) \\ u(v_3) = 1\angle\big(\theta_{j+1} + \pi + 0.5\big|\theta_j - \theta_{j+1} + \frac{\pi}{2}\big|\big) \end{cases}$$

To have the edge E_r^+ as the exit edge, the control label is C_r^+ and the control values at the vertices are:

$$\begin{cases} u(v_0) = u(v_1) = 1\angle\big(\theta_j + 0.5\big|\theta_j - \theta_{j+1} + \frac{\pi}{2}\big|\big) \\ u(v_2) = u(v_3) = 1\angle\big(\theta_{j+1} - 0.5\big|\theta_j - \theta_{j+1} + \frac{\pi}{2}\big|\big) \end{cases}$$

To have the edge E_r^- as the exit edge, the control label is C_r^- and the control values at the vertices are:

$$\begin{cases} u(v_0) = u(v_1) = 1\angle\big(\theta_j + \pi - 0.5\big|\theta_j - \theta_{j+1} + \frac{\pi}{2}\big|\big) \\ u(v_2) = u(v_3) = 1\angle\big(\theta_{j+1} + \pi + 0.5\big|\theta_j - \theta_{j+1} + \frac{\pi}{2}\big|\big) \end{cases}$$

To have the edge E_θ^+ as the exit edge, the control label is C_θ^+ and the control values at the vertices are:

$$\begin{cases} u(v_0) = 1\angle\big(\theta_j + 0.5\big|\theta_j - \theta_{j+1} + \frac{\pi}{2}\big|\big) \\ u(v_1) = 1\angle\big(\theta_j + \pi - 0.5\big|\theta_j - \theta_{j+1} + \frac{\pi}{2}\big|\big) \\ u(v_2) = 1\angle\big(\theta_{j+1} + 0.5\big|\theta_j - \theta_{j+1} + \frac{\pi}{2}\big|\big) \\ u(v_3) = 1\angle\big(\theta_{j+1} + \pi - 0.5\big|\theta_j - \theta_{j+1} + \frac{\pi}{2}\big|\big) \end{cases}$$

To have the edge E_θ^- as the exit edge, the control label is C_θ^- and the control values at the vertices are:

$$\begin{cases} u(v_0) = 1\angle\big(\theta_j - 0.5\big|\theta_j - \theta_{j+1} + \frac{\pi}{2}\big|\big) \\ u(v_1) = 1\angle\big(\theta_j + \pi + 0.5\big|\theta_j - \theta_{j+1} + \frac{\pi}{2}\big|\big) \\ u(v_2) = 1\angle\big(\theta_{j+1} - 0.5\big|\theta_j - \theta_{j+1} + \frac{\pi}{2}\big|\big) \\ u(v_3) = 1\angle\big(\theta_{j+1} + \pi + 0.5\big|\theta_j - \theta_{j+1} + \frac{\pi}{2}\big|\big) \end{cases}$$

Now, the responsibility of the actuator is to relate the discrete symbol $u_d \in \{C_0, C_r^-, C_r^+, C_\theta^+, C_\theta^-\}$ to the continuous control signal $u_c(x)$. Using the properties of multi-affine functions as described in Proposition 1, the control signal can be constructed as

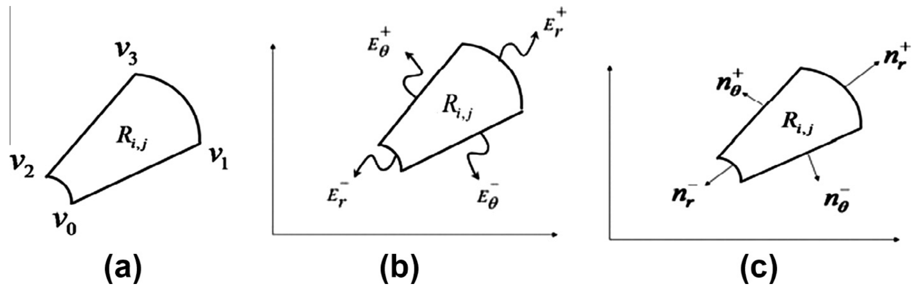

Fig. 3. (a) Vertices of the element $R_{i,j}$. (b) Edges of the element $R_{i,j}$. (c) Outer normals of the element $R_{i,j}$.

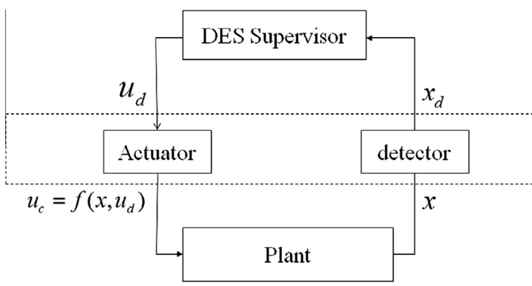

Fig. 4. Linking the discrete supervisor to the plant via an interface layer.

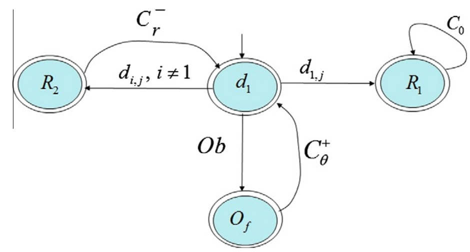

Fig. 5. The formation supervisor.

$u_c(x) = f(x, u_d) = \sum_{m=0}^{3} \lambda_m(x) u(v_m)$, where $u(v_m)$, $m = 0, \ldots, 3$, are the above listed control values at the vertices corresponding to the control label u_d.

4.2. The supervisor layer

Using these control labels, a discrete supervisor is designed for a follower UAV involved in a formation mission. In this supervisor, shown in Fig. 5, when a detection event $d_{i,j}$ appears, the supervisor will be informed that the system has entered the new region $R_{i,j}$. If the detection event is $d_{1,j}$, it means that the system has entered the first circle of the partitioned space and the formation is achieved. Hence, to keep the formation, the system should remain in this region for the rest of the mission. In this case, keeping the formation can be done by activating the controller C_0. If the trajectory has not reached one of the partitions in the first circle ($i > 1$), then the event C_r^- should be activated to move towards the origin. Meanwhile if the leader is located on the way of the follower towards the origin, the event *Ob* will be generated which alarms the supervisor about the collision. To avoid the collision, it is sufficient to drive the follower's path to turn anticlockwise and then, resume the mission. Hence, after observing the event *Ob*, the supervisor activates the event C_θ^+.

5. Implementation issues

5.1. Smooth control

When the system trajectory enters a new region, a new discrete command will be generated. This may cause the discontinuity in the generated control signal to be applied to the lower levels of the control structure. For example, Fig. 6 shows a case that the control command C_r^- has pushed the system's trajectory to transit from the region R_1 to the region R_2. After reaching the region R_2, the control command has changed from C_r^- to C_θ^+. Since the generated continuous control signal is a multi-affine function, based on Corollary 1, the control value at any point on the edges is determined by the control values at its vertices. In this example, $u(v_0(R_1)) = u(v_1(R_2))$ but $u(v_2(R_1)) \neq u(v_3(R_2))$. Since, the control values at the vertices of the common edge between R_1 and R_2 changes, there is a jump on the generated continuous

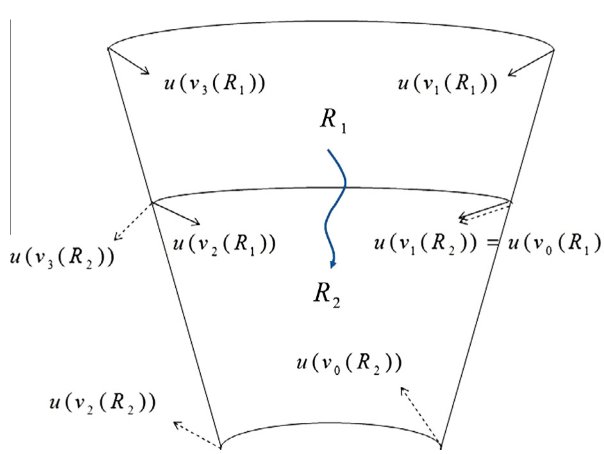

Fig. 6. The control values at the vertices when the system trajectory transits from region R_1 to region R_2 and the discrete command changes from C_r^- to C_θ^+.

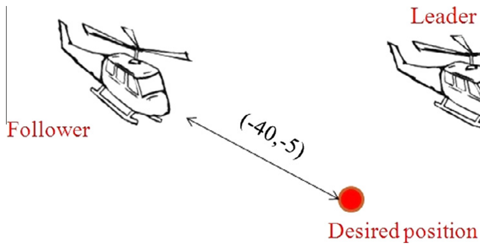

Fig. 7. The schematic of the scenario with a real follower and a virtual fixed leader (Scenario 1).

control signal. Next theorem shows how we can resolve this problem.

Theorem 3. *Let the command C_q^s steers the system's trajectory from the region $R_{i,j}$ to the region $R_{i',j'}$ and then, the supervisor issues the new command $C_{q'}^{s'}$. For this transition, the multi-affine controller $u(x)=\sum_{v_m\in V_c}\lambda_m\hbar(u(v_m)_{new},u(v_m)_{old})+\sum_{v_m\in V_n}\lambda_m(u(v_m))$ provides a smooth control signal, and drives all the system's trajectories to exit from the exit edge $E_{q'}^{s'}$. Here λ_m, m = 0, 1, 2, 3, are given in Proposition 1, V_n is the set of vertices whose control values do not change due to the transition, and V_c is the set of vertices whose control values change after the system's trajectory enters the region $R_{i',j'}$. For these vertices, $u(v_m)_{old}$ and $u(v_m)_{new}$ are the control values at the vertex v_m before and after transiting to $R_{i',j'}$, respectively. The function $\hbar$provides a smooth rotation from $u(v_m)_{old}$ to $u(v_m)_{new}$ and it can be presented as*

$$\hbar(u(v_m)_{new},u(v_m)_{old})=\begin{cases} r_m\angle(\frac{t}{\Delta t}\theta_{m_{new}}+(1-\frac{t}{\Delta t})\theta_{m_{old}}) & t<\Delta t \\ r_m\angle\theta_{m_{new}} & t\geqslant\Delta t \end{cases}$$

where $u(v_m)_{new}=r_m\angle\theta_{m_{new}}$, $u(v_m)_{old}=r_m\angle\theta_{m_{old}}$. Also, Δt is the transition time.

Proof. Let $C_q^s=C_r^-$ and $C_{q'}^{s'}=C_\theta^+$. As shown in Fig. 6, for this sequence of control commands, after transiting from $R_{i,j}$ to $R_{i',j'}$, the control value at the vertex v_3 changes from $u(v_3)_{old}$ to $u(v_3)_{new}$, and for the other vertices v_m, m = 0, 1, 2, there is no jump on the control values.

From the definition of the transition rule, $\hbar$, since for the whole transition time, the control values at the vertices satisfy the conditions of Theorem 1, the system's trajectory cannot leave the region through the non-exit edges $E_{0,2}$, $E_{0,1}$, $E_{1,3}$. Also, at the

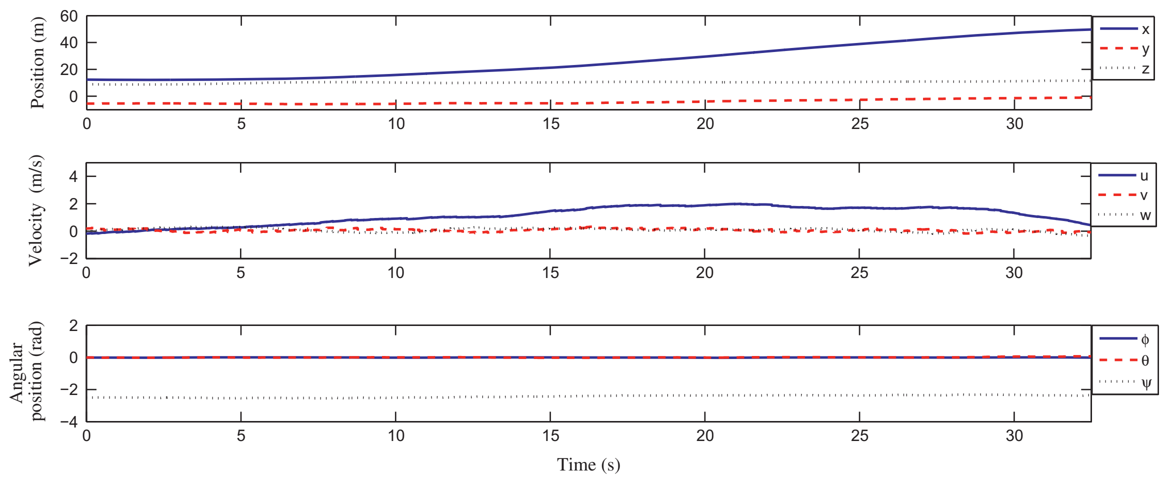

Fig. 8. The state variables of the follower in Scenario 1.

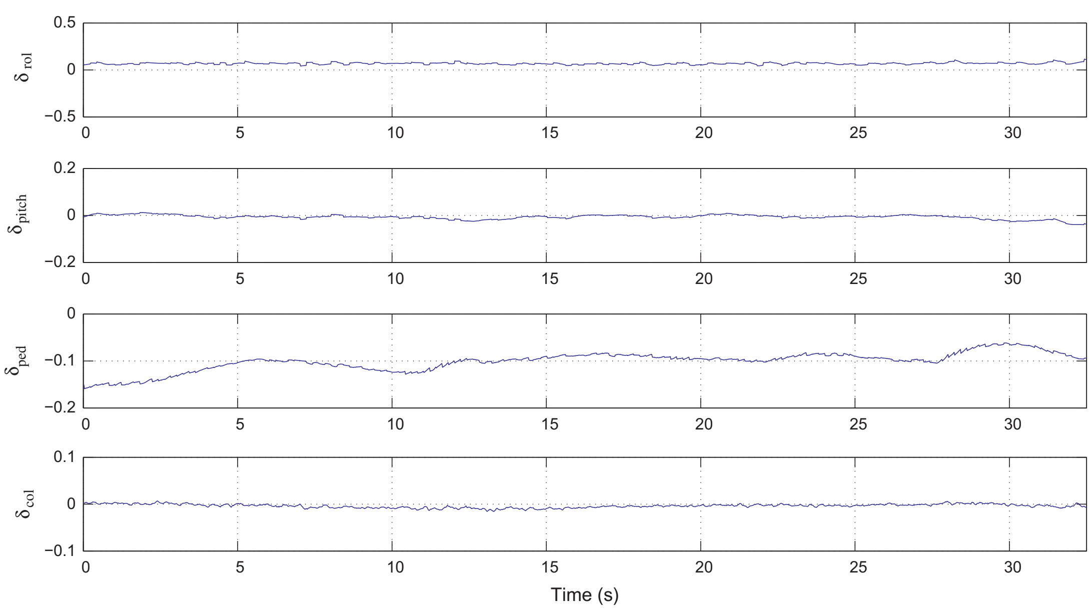

Fig. 9. Control signals of the follower UAV in Scenario 1.

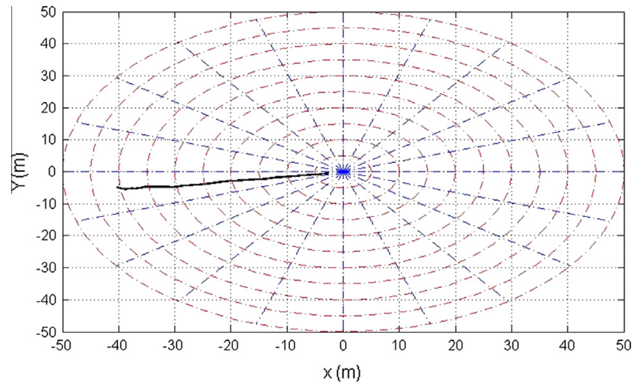

Fig. 10. The leader position in the relative frame in Scenario 1.

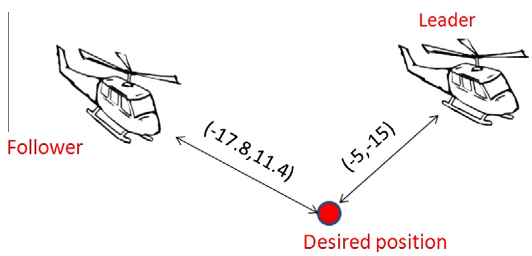

Fig. 11. The schematic of the scenario with for a leader–follower case tracking a line (Scenario 2).

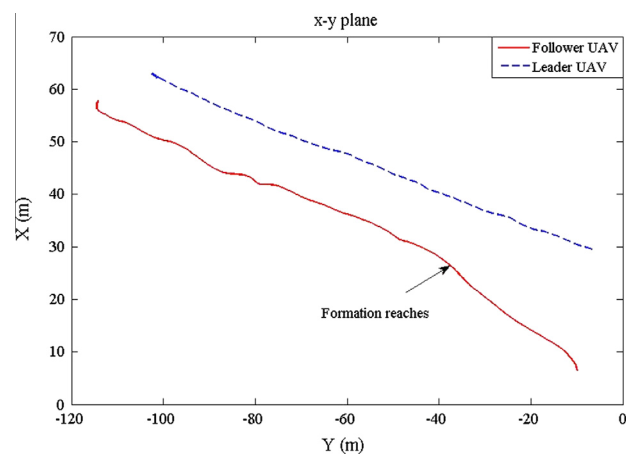

Fig. 12. The position of the UAVs in the x–y plane in Scenario 2.

beginning of the transition mode, the control values at the vertex v_3 does not satisfy the conditions of Theorem 2, and hence, it cannot be concluded that the system's trajectory leaves the region through $E_{2,3}$. But, at some time, $u(v_3)$ will eventually reach $u(v_3)_{new}$, and the configuration of the vector field at the vertices will satisfy the conditions of Theorem 2 so that it can be guaranteed that the system's trajectory for sure leaves the region $R_{i',j'}$ through the edge $E_{2,3}$, while there is no jump at the value of the control signal due to the smooth transition of the control values at the vertices. The same reasoning can be done for the other sequences of the control commands. □

Remark 2. In [14], it was shown that the polar abstracted model is bisimialr to the original system meaning that for any transition in the abstracted model, there is a transition in the original system and vice versa. From Theorem 3, it can be immediately concluded that the result is also valid for the case of smooth transition mechanism. This is due to the fact that based on Theorem 3, all of the trajectories finally will leave the region through the desired exit edge and the smooth transition mechanism does not let the system's trajectories to exit from non-exit edges, leading to the following corollary:

Corollary 3.

The smooth transition mechanism introduced in Theorem 3 preserves the bisimilarity relation between the abstracted model and the original hybrid system.

5.2. Time sequencing of the events

In the proposed framework, we assume that the discrete control signals, C_0, C_r^+, C_r^-, C_θ^+, or C_θ^-, can be applied after entering a new region, unless a collision alarm be generated which requires an immediate reaction. But, the question is that, transiting to a new region, when should exactly the new control signals be applied to the system?.

Indeed, from practical reasons, the detector cannot recognize entering a region until the system trajectory crosses the region's boundary. This is why in the definition of the exit edge we have considered a time delay $\tau_d > 0$. Only after this time delay, the controller can be ensured that the system trajectory has transited to a new region and hence, a new actuation event C_0, C_r^+, C_r^-, C_θ^+, or C_θ^-, can be generated based on the desired behavior. The time delay, $\tau_d > 0$, could be very small but cannot be zero. This guarantees that the resulting model is not Zeno [22], meaning that the number of discrete transitions in a finite time is finite.

More precisely, as described in Section 4.2, when the last visited region is $R_{i,j}$ and the supervisor detects an event $d_{i',j'}$, it means that the system trajectory has entered the new region $R_{i',j'}$. Then, a control command C_q^s will be generated which pushes the system trajectory to enter another region $R_{i'',j''}$. Again, when the system's trajectory crosses the boundaries of the region $R_{i'',j''}$, this will cause the event $d_{i'',j''}$ to appear. Hence, for the successive events $d_{i',j'}$, C_q^s, $d_{i'',j''}$, we will have:

$$t(d_{i',j'}) < t(C_q^s) < t(d_{i'',j''}). \quad (8)$$

To consider the time delay $\tau_d > 0$, the sequence of the events should respect the following condition:

$$t(C_q^s) \geqslant t(d_{i',j'}) + \tau_d. \quad (9)$$

6. Implementation results

To verify the algorithm, we have conducted several flight tests. In the first scenario, to monitor reaching the formation behavior of the UAVs, the follower should reach the desired position with

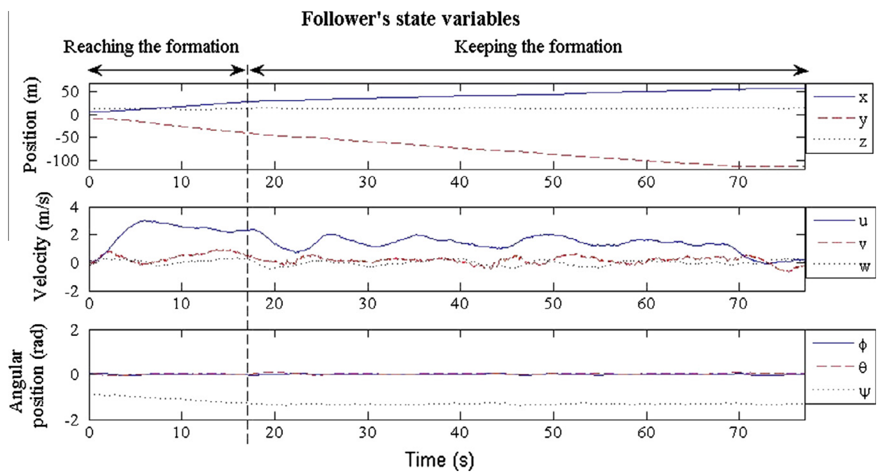

Fig. 13. The state variables of the follower in Scenario 2.

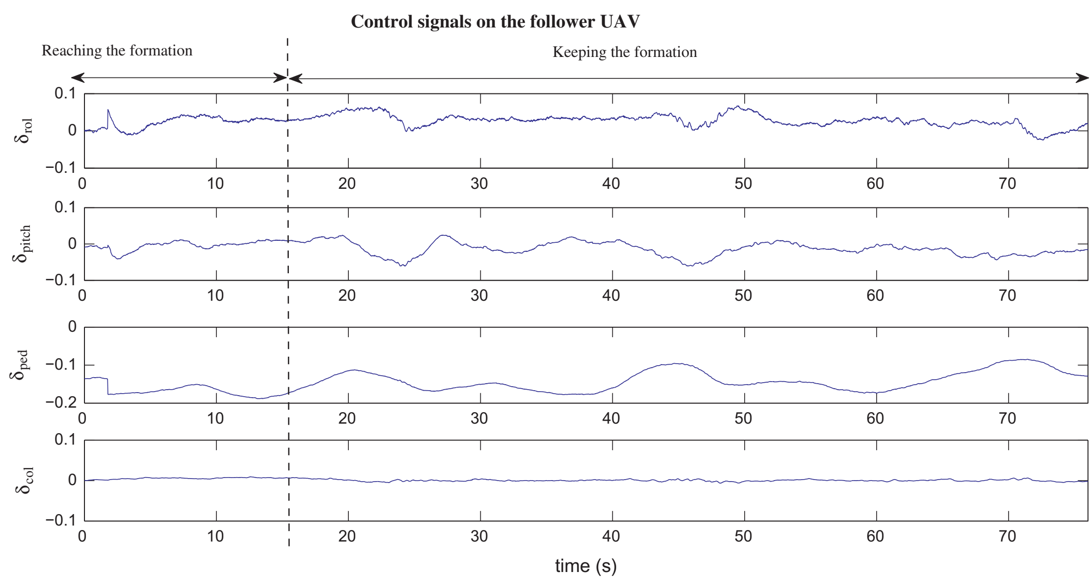

Fig. 14. Control signals of the follower UAV in Scenario 2.

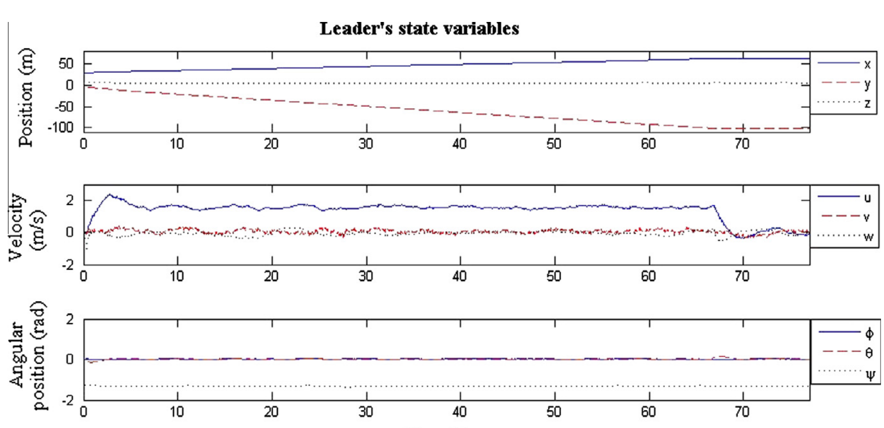

Fig. 15. The state variable of the leader in Scenario 2.

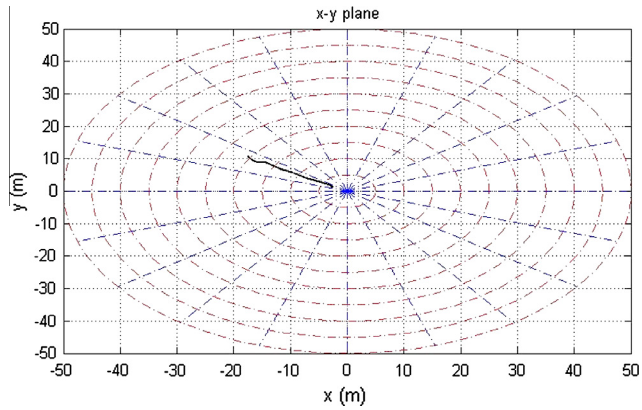

Fig. 16. The distance of the follower from the desired position in Scenario 2.

respect to a fixed leader. In this test the control horizon R_m = 50 m, n_r = 10, and n_θ = 20. The follower is initially located at a point which has a relative distance of $(dx, dy) = (-40, -5)$ with respect to the desired position as shown in Fig. 7. The follower state variables and control signals are shown in Fig. 8 and Fig. 9, respectively. The follower UAV position in the relative frame is shown in Fig. 10. As it can be seen the follower UAV has started from the region $R_{9,11}$ and finally has reached the region $R_{1,11}$ which is located in the first circle and hence, the formation has been achieved.

In the second scenario, to monitor how the follower is able to maintain the achieved formation, the leader tracks a line path, and the follower should reach and keep the formation. In this test, the control horizon R_m is 50 m, n_r = 10, and n_θ = 20. The follower is initially located at a point which has a relative distance of $(dx, dy) = (-17.8, 11.4)$ with respect to the desired position and the distance between the desired position and the leader is $(dx, dy) = (-5, -15)$ as shown in Fig. 11.

The position of the UAVs in x–y plane is shown in Fig. 12. The follower state variables and control signals are shown in Figs. 13 and 14, respectively. The state variables of the leader are shown in Fig. 15. The relative distance of the follower UAV from the desired position is shown in Fig. 16. As it can be seen the follower UAV has finally reached the first circle after 17 s and then, it has been able to maintain the formation.

In the third flight test, the leader path is a circle which is a more complex path. Here, the control horizon R_m is 50 m, n_r = 10, and n_θ = 20. The follower is initially located at a point which has a relative distance of $(dx, dy) = (-30.5, 13.2)$ with respect to the desired position and the distance between the desired position and the leader is $(dx, dy) = (-5, -15)$ as shown in Fig. 17. In this test the leader tracks a circle path with a diameter of 40 m. After a while, the follower reaches the formation and can keep it for the rest of the mission. The position of the UAVs in x–y plane is shown in Fig. 18. The follower state variables and control signals are shown in Figs. 19 and 20, respectively. The state variables of the leader during the mission is shown in Fig. 21. The relative distance of follower UAV from the desired position is shown in Fig. 22. As it can be seen the follower UAV has finally reached the first circle and the formation has been achieved. The video for the second and third experiments is available at uav.ece.nus.edu.sg/video/2dHybridFormation.mpg.

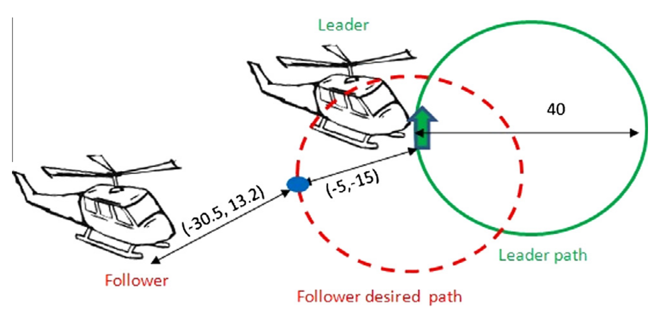

Fig. 17. The schematic of the scenario with for a leader–follower case tracking a circle (Scenario 3).

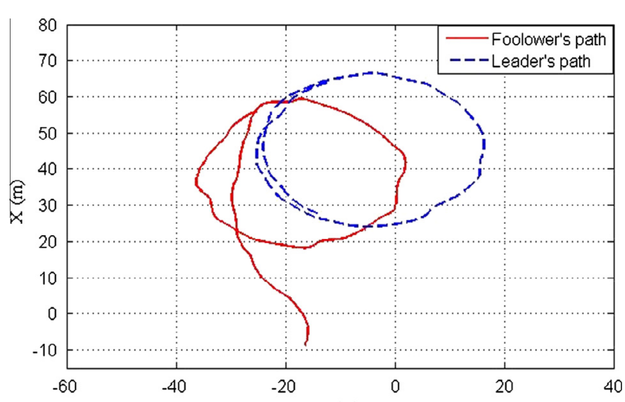

Fig. 18. The position of the UAVs in the x–y plane in Scenario 3.

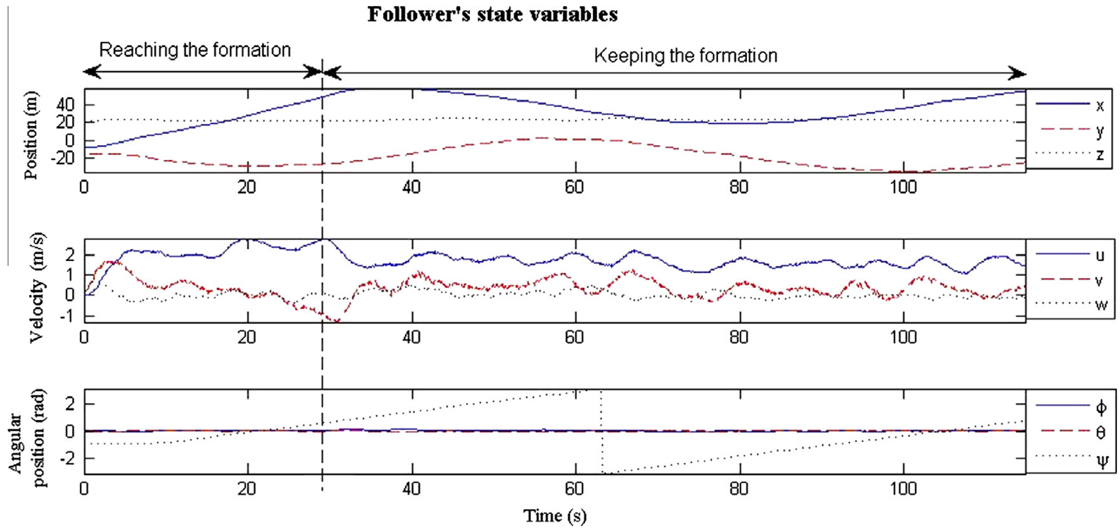

Fig. 19. The state variables of the follower in Scenario 3.

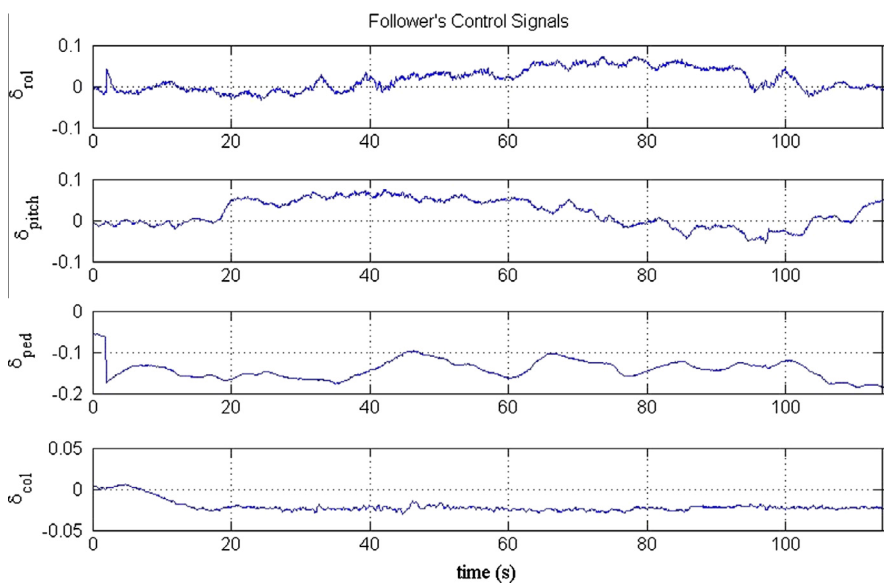

Fig. 20. Control signals of the follower UAV in Scenario 3.

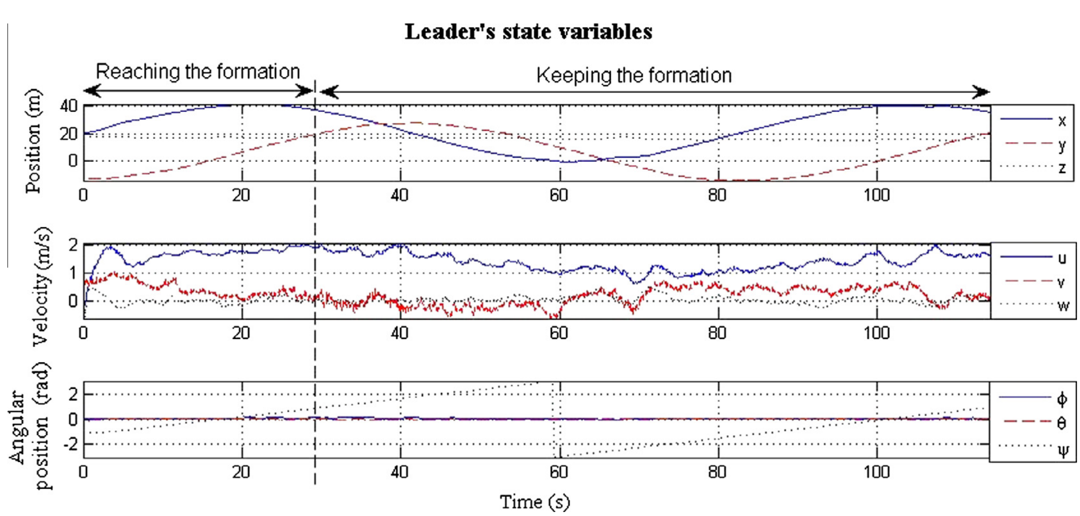

Fig. 21. The state variables of the leader in Scenario 3.

6.1. Extension to the 3-D space

In [23], the result is extended to the 3-D space. For the 3-D case, the DES model is different and accordingly, the designed supervisor need to be redesign; however, the procedure for the design and implementation of the supervisor is similar to what was discussed here. For this case, a flight test is conducted in which the initial relative distance between the follower and the desired position is $(dx, dy, dz) = (-16.1, 22.5, -14.7)$, and the distance between the desired position and the desired position is $(dx, dy, dz) = (15, 10, 10)$. The UAVs' position are shown in Fig. 23. The projection of the relative distance onto the x–y plane is shown in Fig. 24. In this experiment,

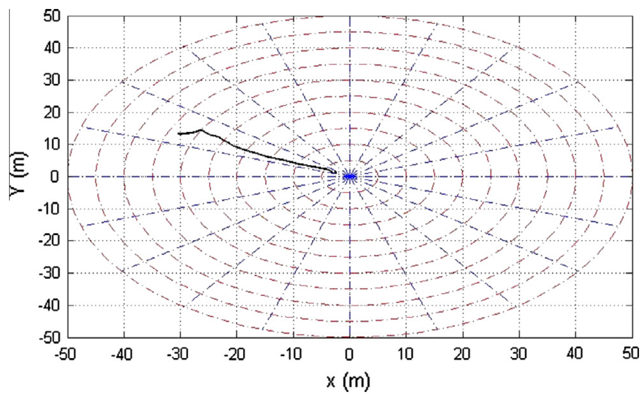

Fig. 22. The distance of the follower from the desired position in Scenario 3.

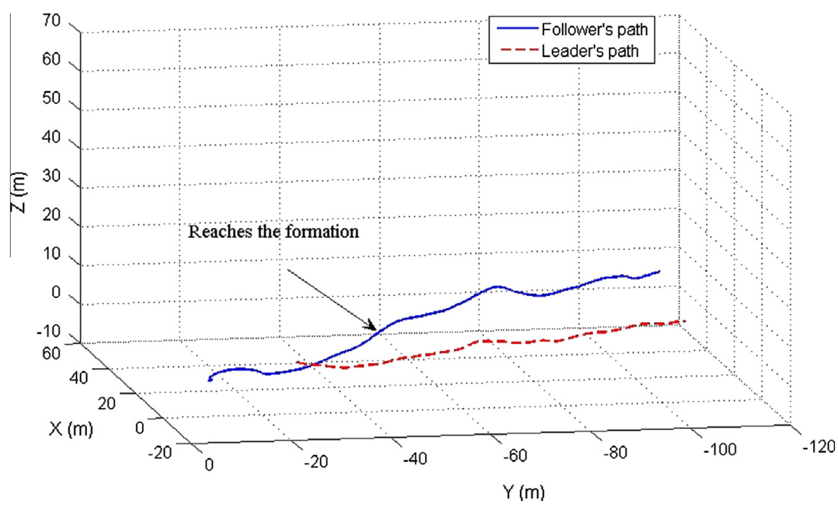

Fig. 23. The position of the UAVs in the actual flight test in Scenario 3.

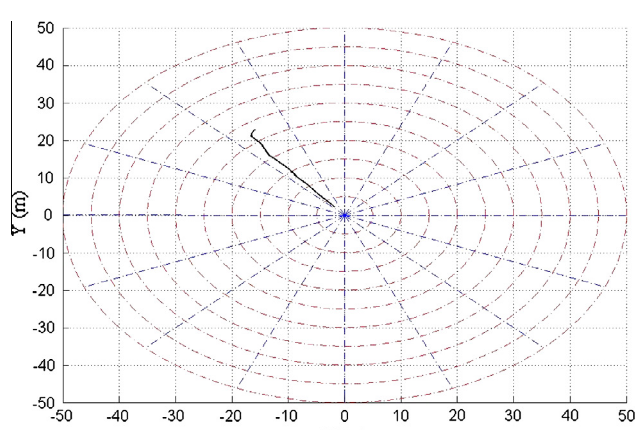

Fig. 24. The relative distance between the UAVs projected onto x–y plane in Scenario 3.

after a while, the formation has been reached and it has been successfully maintained. A video of this experiment is available at: uav.ece.nus.edu.sg/video/hybridformation.mpg.

7. Conclusion

In this paper a bumpless hybrid supervisory control algorithm was applied to the formation control of the UAVs. The method was based on polar abstraction of the motion space and the use of properties of multi-affine functions over the partitioned space. The implementation issues for this control method were investigated. To implement the algorithm, an interface layer was introduced which connects the discrete supervisor to the regulation layer of the UAV. This interface layer is composed of two main blocks: the detection block to generate the detection events based on the plant continuous signals; and the actuator block to convert discrete commands of the supervisor to a continuous form, applicable to the plant. Also, a method was introduced to smoothly generate control signals during the transition through the partitioned regions. The implementation issues were discussed in details. Several actual flight tests were conducted to verify the algorithm. The proposed formation algorithm can be extended to a multi-follower case, however, it is required to develop a more sophisticated collision avoidance mechanism as we will consider this issue as the future direction of this research.

Acknowledgment

The authors thank Dr. Xiangxu Dong, Dr. Guowei Cai, and Dr. Feng Lin for their technical support during the implementations and flight tests.

References

[1] Anderson B, Fidan B, Yu C, Walle D. UAV formation control: theory and application. In: Blondel V, Boyd S, Kimura H, editors. Recent advances in learning and control. Lecture notes in control and information sciences. Berlin/Heidelberg: Springer; 2008.
[2] van der Walle D, Fidan B, Sutton A, Yu C, Anderson D. Non-hierarchical uav formation control for surveillance tasks, In: American Control Conference, 2008. p. 777–782..
[3] Hu J, Xu J, Xie L. Cooperative search and exploration in robotic networks, in Unmanned Systems, 2013;1(1):121–42.
[4] How J, King E, Kuwata Y. Flight demonstrations of cooperative control for UAV teams. In: AIAA 3rd unmanned unlimited technical conference, 2004.
[5] De Gennaro M, Jadbabaie A. Formation control for a cooperative multi-agent system using decentralized navigation functions. In: American control conference, 2006.
[6] Paul T, Krogstad T, Gravdahl J. Modelling of UAV formation flight using 3d potential field. Simul Modell Practice Theory 2008;16(9):1453–62.
[7] Hassan G, Yahya K, ul Haq I. Leader–follower approach using full-state linearization via dynamic feedback. In: International conference on emerging technologies, 2006. p. 297–305.
[8] Shames I, Fidan B, Anderson BD. Minimization of the effect of noisy measurements on localization of multi-agent autonomous formations. Automatica 2009;45(4):1058–65.
[9] Linorman N, Liu H. Formation UAV flight control using virtual structure and motion synchronization. In: American control conference. IEEE; 2008. p. 1782–7.
[10] Zamani M, Lin H. Structural controllability of multi-agent systems. In: American control conference. 2009. p. 5743–8.
[11] Jansson J, Gustafsson F. A framework and automotive application of collision avoidance decision making. Automatica 2008;44(9):2347–51.
[12] Schlanbusch R, Kristiansen R, Nicklasson PJ. Spacecraft formation reconfiguration with collision avoidance. Automatica 2011;47(7):1443–9.
[13] Cetin B, Bikdash M, Hadaegh F. Hybrid mixed-logical linear programming algorithm for collision-free optimal path planning. Control Theory Appl, IET 2007;1(2):522–31.
[14] Karimoddini A, Lin H, Chen BM, Lee TH. Hybrid formation control of the unmanned aerial vehicles. Mechatronics 2011;21(5):886–98.
[15] Koutsoukos X, Antsaklis P, Stiver J, Lemmon M. Supervisory control of hybrid systems. Proc IEEE 2000;88(7):1026–49.
[16] Tabuada P. Verification and control of hybrid systems: a symbolic approach. New York Inc.: Springer-Verlag; 2009.
[17] Alur R, Henzinger T, Lafferriere G, Pappas G. Discrete abstractions of hybrid systems. Proc IEEE 2000;88(7):971–84.
[18] Ramadge P, Wonham W. The control of discrete event systems. Proc IEEE 1989;77(1):8–98.
[19] Cai G, Chen BM, Peng K, Dong M, Lee TH. Modeling and control system design for a UAV helicopter. In: 14th IEEE Mediterranean conference on control and automation, 2006. p. 1–6.
[20] Karimoddini A, Cai G, Chen BM, Lin H, Lee TH. Hierarchical control design of a UAV helicopter. In: Advances in flight control systems, INTECH, Vienna, Austria; 2011.
[21] Karimoddini A, Cai G, Chen BM, Lin H, Lee TH. Multi-layer flight control synthesis and analysis of a small-scale UAV helicopter. In: IEEE conference on robotics automation and mechatronics, 2010. p. 321–6.
[22] Johansson KH, Egerstedt M, Lygeros J, Sastry S. On the regularization of Zeno hybrid automata, systems and amp. Control Lett 1999;38(3):141–50.
[23] Karimoddini A, Lin H, Chen BM, Lee TH. Hybrid three-dimensional formation control for unmanned helicopters. Automatica 2012;49(2):424–33.