International Journal of Systems Science
Vol. 40, No. 5, May 2009, 539–552

Taylor & Francis
Taylor & Francis Group

Distributed network-based formation control

Dongbing Gu* and Huosheng Hu

Department of Computing and Electronic Systems, University of Essex, Wivenhoe Park, Colchester, CO4 3SQ, UK

(*Received 15 October 2007; final version received 5 December 2008*)

This article investigates a formation control problem of mobile robots. Two main issues of the formation control problem are addressed; one is the implementation of distributed formation control and another is the analysis of effect of network communication on the closed-loop systems. The distributed formation control is obtained by modelling the formation control problem as a linear-quadratic Nash differential game through the use of graph theory. The obtained distributed controller is a state feedback controller that requires both local state and neighbour robot states. As the neighbour robot states are obtained via networks, the network performance affects the control system. Network-induced delay is considered in this article. The stability analysis of the proposed distributed formation controller under network-induced delay is given. A compensation scheme for network-induced delay is proposed and analysed. Mobile robots with double integrator dynamics are used in the formation control simulation. Simulation results are provided to justify the models and solutions.

Keywords: robotics; decentralised control; distributed formation control; linear-quadratic differential game

1. Introduction

Team formation is one of the salient features of multiple mobile agent systems. It is inspired by swarming behaviour of living beings, such as flocks of birds, schools of fish, herds of wildbeast, and colonies of bacteria. It is known that such a cooperative biological behaviour has certain advantages, including avoiding predators, increasing the chance of finding food, saving energy (Weimerskirch, Martin, Clerquin, Alexandre and Jiraskova 2001), etc. Engineering applications of the team formation include automated highway systems (Swaroop and Hedrick 1996; Pant, Seiler and Hedrick 2002), cooperative robot reconnaissance (Balch and Arkin 1998), manipulation operation (Tanner, Loizou and Kyriakopoulos 2003), formation flight control (Giulietti, Pollini and Innocenti 2000; Raffard, Tomlin and Boyd 2004; Stipanovic, Inalhan, Teo and Tomlin 2004), satellite clustering (McInnes 1995), distributed sensor networks (Ogren, Egerstedt and Hu 2002), etc.

Formation control has been studied in robotics within different structures, such as behaviour-based structure, leader–follower structure and virtual leader structure. The formation control in the behaviour-based structure is achieved by building up a group of formation related behaviours (Balch and Arkin 1998). It is suitable for uncertain environments, but lack of a rigorous theoretic analysis. The leader–follower structure is constructed by a string of chain where each robot follows one or two robots (Desai, Ostrowski, and Kumar 2001; Fierro, Das, Kumar and Ostrowski 2001; Das et al. 2002). The chain structure leads to a poor disturbance rejection property. The virtual leader structure constitutes fictitious leaders that represent the desired robot positions. Using receding horizon approaches for the virtual leader structure has been investigated, such as using contractive stability constraint in Camponogara, Jia, Krogh and Talukdar (2002) and compatibility constraint in Dunbar and Murray (2006).

Currently, formation control uses a common team objective as a formation objective, i.e. all the individual robot's interests are added together to form an identical team objective. However, individual robots in a team may have different interests. Game theory is an effective tool to model a team with different interests. One of the main issues we want to address in this article is the implementation of distributed formation control. This can be implemented by using different cost functions for different robots. As these cost functions are only related to neighbour robots, not the entire team, distributed formation control can be obtained by analysing the differential game problem.

Formation control cost functions can be casted as a linear-quadratic form by using graph theory. The formation control problem is converted to the

*Corresponding author. Email: dgu@essex.ac.uk

ISSN 0020–7721 print/ISSN 1464–5319 online

DOI: 10.1080/00207720902750029
http://www.informaworld.com

coupled (asymmetric) Riccati differential equation problem. The open-loop information structure is based on the assumption that the only information robots have is their present states. It can be interpreted as such the robots simultaneously determine their strategies at the beginning of the game and use this open-loop solution for the whole period of the game (Abou-Kandil, Freiling, Ionescu and Jank 2003; Engwerda 2005). The finite horizon open-loop Nash equilibrium can be combined with a receding horizon approach to produce a resultant receding horizon Nash control.

In the distributed control, neighbour robots exchange their information with each other via wireless networks. The entire team is modelled as a networked control system. It is well known that communication networks can degrade control performance of networked control systems due to the network-induced delay, packet loss, etc (Nilsson 1998; Zhang, Brannicky and Philips 2001). Currently, research of networked control systems concentrates on systems with a single controller where communication is necessary in the feedback channel (between sensors and controller) and forward channel (between controller and actuator) (Walsh, Beldiman and Bushnell 2001; Zhang et al. 2001; Park, Kim, Kim and Kwon 2002; Nesic and Teel 2004; Liu, Xia, Chen, Rees and Hu 2007). The problems raised by networked control systems include network-induced delay, packet loss and multiple packet transmission (Zhang et al. 2001). The network-induced delay can be constant or random. For the delay less than one sample period, the stability analysis was provided in Nilsson (1998) and Walsh et al. (2001). The notion of maximum allowable transfer interval (MATI) was introduced in Walsh et al. (2001), to represent the maximum separation interval between successive sensor data for a stabilising system. The compensation for delay less than one sample period was discussed in Zhang et al. (2001), based on a linear estimator. The extension of MATI from linear systems to non-linear systems with disturbance was given in Nesic and Teel (2004) by using input–output stability properties. For linear systems, the compensation for constant and random delay longer than one sample period was investigated in Liu et al. (2007).

In this article, we will concentrate on the network delay problem. As robots only transmit state information between neighbours, we assume only one packet for each state transmission. This can avoid the problem caused by multiple packet transmission, assuming the delay in the forward and feedback channels is negligible as robots directly connect to sensors and actuators without using networks and the interfaces are very fast comparing with the sample period. The network-induced delay only occurs between communication of neighbouring robots. Obtaining a constant network delay is one of the major objectives in networked systems. In Roedig, Barroso and Sreenan (2006), a so-called *f*-MAC protocol was designed in order to have a deterministic behaviour for network-based systems. In this article, constant network-induced delay will be investigated, which may be longer than one sample period. A compensation scheme for the delay is proposed for the distributed formation control.

In the following, the formation control problem is stated in Section 2. We will address two important issues in the formation control problem in the following two sections. One is the implementation of distributed control in Section 3 and another is the analysis of network-induced delay and compensation in Section 4. The simulation results are provided in Section 5. Finally, conclusions and future work are given in Section 6.

2. Formation control

2.1. *Robot model*

A team has m robots, each of which is described by its dynamics. For robot i with n-dimensional coordinates $q^i \in \mathbb{R}^n$, the state and control vectors are $z_t^i = [(q_t^i)^T, (\dot{q}_t^i)^T]^T \in \mathbb{Z}$ and $u_t^i \in \mathbb{U}$ $(i = 1, \ldots, m)$. The robot dynamics is:

$$\dot{z}_t^i = az_t^i + bu_t^i, \tag{1}$$

where

$$a = \begin{bmatrix} 0 & I_{(n)} \\ 0 & 0 \end{bmatrix}, \; b = \begin{bmatrix} 0 \\ I_{(n)} \end{bmatrix}.$$

The matrix $I_{(n)}$ is the identity matrix of dimension n.

By concatenating the states of all m robots in a team into a vector $z = [(z^1)^T, \ldots, (z^m)^T]^T \in \mathbb{R}^{2nm}$, the team dynamics is:

$$\dot{z}_t = Az_t + \sum_{i=1}^{m} B^i u_t^i = Az_t + Bu_t, \quad t \geq 0 \tag{2}$$

where $A = I_{(m)} \otimes a$, $B^i = [0, \ldots, 1, \ldots, 0]^T \otimes b$, and $B = [B^1, B^2, \ldots, B^m]$. The operator $\otimes$ is the Kronecker product.

Let $z^{i,r} = [(q^{i,r})^T, (\dot{q}^{i,r})^T]^T$ be the desired state vector for robot i. The desired team state vector is then represented by $z^r = [(z^{1,r})^T, \ldots, (z^{m,r})^T]^T \in \mathbb{R}^{2nm}$. The desired state $z^{i,r}$ should also have the same dynamics with the robot dynamics (1):

$$\dot{z}_t^{i,r} = az_t^{i,r} + bu_t^{i,r}. \tag{3}$$

And the concatenating state equation is:

$$\dot{z}_t^r = Az_t^r + \sum_{i=1}^{m} B^i u_t^{i,r}, \quad t \geq 0. \tag{4}$$

2.2. *Formation model*

A graph can be used to represent the formation control interconnection between robots. A vertex of the graph corresponds to a robot and the edges of the graph capture the dependence of the interconnections. Formally, a directed graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ consists of a set of vertices $\mathcal{V} = \{v^1, \ldots, v^m\}$, indexed by the robots in a team, and a set of edges $\mathcal{E} = \{(v^i, v^j) \in \mathcal{V} \times \mathcal{V}\}$, containing ordered pairs of distinct vertices. Assuming that the graph has no loops, i.e. $(v^i, v^j) \in \mathcal{E}$ implies $v^i \neq v^j$. A graph is connected if for any vertices $(v^i, v^j) \in \mathcal{V}$, there exists a path of edges in $\mathcal{E}$ from v^i to v^j. An edge-weighted graph is a graph in which each edge is assigned a weight. The edge (v^i, v^j) is associated with weight $\omega^{ij} \geq 0$.

The incidence matrix D of a directed graph $\mathcal{G}$ is the $\{0, \pm 1\}$-matrix with rows and columns indexed by vertices of $\mathcal{V}$ and edges of $\mathcal{E}$, respectively, such that the uv-th entry of D is equal to 1 if the vertex u is the head of the edge v, -1 if the vertex u is the tail of the edge v and 0 otherwise. If graph $\mathcal{G}$ has m vertices and $|\mathcal{E}|$ edges, then incidence matrix D of the graph $\mathcal{G}$ has order $m \times |\mathcal{E}|$ (Godsil and Royle 2001).

The cohesion and separation of formation control are defined by the desired distance vector $d_{ij}^r = z^{i,r} - z^{j,r}$ between two neighbours v^i and v^j. The formation error vector is defined as $z^i - z^j - d_{ij}^r$ for edge (v^i, v^j). Let $\hat{D} = D \otimes I_{(2n)}$. From the definition of the incidence matrix, we know the whole team formation error can be expressed in a matrix form:

$$\begin{aligned}\sum_{(i,j)\in\mathcal{E}} \omega^{ij} \| z^i - z^j - d_{ij}^r \| &= (z - z^r)^T \hat{D}\hat{W}\hat{D}^T (z - z^r) \\ &= \| z - z^r \|^2_{\hat{D}\hat{W}\hat{D}^T},\end{aligned} \tag{5}$$

where $\hat{W} = W \otimes I_{(2n)}$ and $W = \mathrm{diag}[\omega^{ij}]$ is a diagonal weight matrix with dimension $|\mathcal{E}|$. We use $\| z - z^r \|^2_{\hat{D}\hat{W}\hat{D}^T}$ for $(z - z^r)^T \hat{D}\hat{W}\hat{D}^T (z - z^r)$. Following Godsil and Royle (2001), we define the Laplacian of a graph $\mathcal{G}$ as $L = DWD^T$. For a directed graph $\mathcal{G}$, the Laplacian is independent of the choice of graph orientation. The Laplacian L is symmetric and positive semi-definite. We have:

$$\begin{aligned}\hat{L} &= \hat{D}\hat{W}\hat{D}^T = (D \otimes I_{(2n)})(W \otimes I_{(2n)})(D \otimes I_{(2n)})^T \\ &= L \otimes I_{(2n)}.\end{aligned} \tag{6}$$

$\hat{L}$ is also symmetric and positive semi-definite. The team formation error is rewritten as follows:

$$\sum_{(i,j)\in\mathcal{E}} w^{ij} \| z^i - z^j - d_{ij}^r \| = \| z - z^r \|^2_{\hat{L}}. \tag{7}$$

2.3. *Formation control cost function*

The finite horizon cost function of the formation control for robot i can be expressed as follows:

$$\begin{aligned}J^i(u) &= g_T^i(z_T) + \int_0^T C_\tau^i(z_\tau, u_\tau)\mathrm{d}\tau, \\ g_T^i(z_T) &= \sum_{(i,j)\in\mathcal{E}} \omega^{ij} \| z_T^i - z_T^j - d_{ij}^r \|^2, \\ C_\tau^i(z_\tau, u_\tau) &= \sum_{(i,j)\in\mathcal{E}} \mu^{ij} \| z_\tau^i - z_\tau^j - d_{ij}^r \|^2 + \| u^i(\tau) \|^2_{R^i},\end{aligned} \tag{8}$$

where T is the finite time horizon, and $\mu^{ij} \geq 0$, $R^i > 0$, $(i = 1, \ldots, m)$ are the weight parameters. The cost function (8) can be transformed into a standard linear-quadratic form:

$$\begin{aligned}g_T^i(z_T) &= \| z_T - z_T^r \|^2_{K_T^i}, \\ C_\tau^i(z_\tau, u_\tau) &= \| z_\tau - z_\tau^r \|^2_{Q^i} + \| u_\tau^i \|^2_{R^i},\end{aligned} \tag{9}$$

where $K_T^i = \hat{L}_T^i = \hat{D}\hat{W}_T^i\hat{D}^T$, $\hat{W}_T^i = W_T^i \otimes I_{(2n)}$, $W_T^i = \mathrm{diag}[\omega^{ij}]$, $Q^i = \hat{L}^i = \hat{D}\hat{W}^i\hat{D}^T$, $\hat{W}^i = W^i \otimes I_{(2n)}$ and $W^i = \mathrm{diag}[\mu^{ij}]$. K_T^i and Q^i are symmetric and positive semi-definite.

To track a specific trajectory $z^{l,r}$, the leader robot l should track $z^{l,r}$. Thus, the cost function of the leader robot should include a linear-quadratic standard tracking term:

$$\begin{aligned}g_T^l(z_T) &= \| z_T - z_T^r \|^2_{K_T^l} + \| z_T^l - z_T^{l,r} \|^2_{k_T^l} = \| z_T - z_T^r \|^2_{K_T^{\prime l}}, \\ C_\tau^l(z_\tau, u_\tau) &= \| z_\tau - z_\tau^r \|^2_{Q^l} + \| z_\tau^l - z_\tau^{l,r} \|^2_{q^l} + \| u_\tau^l \|^2_{R^l} \\ &= \| z_\tau - z_\tau^r \|^2_{Q^{\prime l}} + \| u_\tau^l \|^2_{R^l},\end{aligned} \tag{10}$$

where $k_T^l = \mathrm{diag}[\omega^l]$, $q^l = \mathrm{diag}[\mu^l]$, $K_T^{\prime l} = K_T^l + \mathrm{diag}[0, \ldots, k_T^l, \ldots, 0]$, and $Q^{\prime l} = Q_l + \mathrm{diag}[0, \ldots, q^l, \ldots, 0]$. $K_T^{\prime l}$ and $Q^{\prime l}$ are also symmetric and positive semi-definite. The leader robot can use $K_T^l = 0$ and $Q^l = 0$, which means the leader robot only tracks the desired trajectory without taking the formation error into account. In such a situation, it is the follower robots who keep the formation by following the leader with a fixed distance.

In the following, the weight matrices in the cost functions are denoted as K_T^i and Q^i for both leader robots or follower robots. From the state Equations (2) and (4) and the cost functions (9) and (10), it can be seen that the formation control is a linear-quadratic tracking problem. By using error state and control vectors, the formation control is viewed as a linear-quadratic regulating problem with z_t as the state vector and u_t as the control vector in the following presentation.

2.4. *Formation control problem*

Each robot in a team can be viewed as a player or decision maker of a differential game. The robot dynamic Equation (2) is the state equation of the differential game with the given initial state z_0 and rewritten as:

$$\dot{z}_t = Az_t + Bu_t, \quad t \geq 0. \tag{11}$$

The cost function J^i defined in (8) is known to each player. The players in the game need to minimise their cost functions in order to find their strategies. Using different cost functions, individual robots can select neighbours and implement distributed controllers. A collection of strategies $\bar{u}^i_t (t \geq 0, i = 1, \ldots, m)$ constitutes a Nash equilibrium if and only if the following inequalities are satisfied for all $u^i_t (t \geq 0, i = 1, \ldots, m)$:

$$\begin{aligned} &J^i(\bar{u}^1, \ldots, \bar{u}^{i-1}, \bar{u}^i, \bar{u}^{i+1}, \ldots, \bar{u}^m) \\ &\quad \leq J^i(\bar{u}^1, \ldots, \bar{u}^{i-1}, u^i, \bar{u}^{i+1}, \ldots, \bar{u}^m), \\ &\quad (i = 1, \ldots, m). \end{aligned}$$

In the open-loop information structure, all players make their decisions based on the initial state z_0. Each player computes its equilibrium strategy at the beginning of the game and no state feedback is available during the whole control period.

To optimise control performance, it is necessary that $\mathbb{U}$ is a compact and convex subset of $\mathbb{R}^n$ containing the origin in its interior, and $\mathbb{Z}$ is a convex, connected subset belonging to $\mathbb{R}^{2n}$ containing $z^{i,r}$ in its interior, for every i. To control a team to keep a formation, it is necessary that graph $\mathcal{G}$ is connected.

3. Distributed formation controller

3.1. *Linear-quadratic open-loop Nash equilibria*

Under the open-loop information structure of a Nash game, the derivation of open-loop Nash equilibria is closely related to the problem of jointly solving m optimal control problem (Basar and Olsder 1995). According to the minimum principle, the conditions for an open-loop Nash equilibrium for two player games are given in Theorem 7.2 in Engwerda (2005). This result is generalised straightforwardly to m player games in Engwerda (2005). We quote this result here for our formation control problem.

Theorem 1: *For a m-robot formation control defined as a finite horizon open-loop Nash differential game by* (11) *and* (12), *let there exist a solution set* $(K^i_t, i = 1, \ldots, m)$ *to the coupled Riccati differential equations*

$$\dot{K}^i_t = -A^T K^i_t - K^i_t A - Q^i + K^i_t \sum_{j=1}^{m} S^j K^j_t, \tag{12}$$

where $S^i = B^i R^{i-1} B^{i^T}$ *and* K^i_T *is the terminal condition. Then the formation control has a unique open-loop Nash equilibrium solution for every initial state as follows*:

$$\bar{u}^i_t = -R^{i^{-1}} B^{i^T} K^i_t \Phi_t z_0, \tag{13}$$

$$\dot{\Phi}_t = \left(A - \sum_{i=1}^{m} S^i K^i_t \right) \Phi_t = A^{\mathrm{cl}}_t \Phi_t, \quad \Phi_0 = I, \tag{14}$$

where $A^{\mathrm{cl}}_t = A - \sum_{i=1}^{m} S^i K^i_t$ *is the closed-loop system matrix. It is easily verified that* $z_t = \Phi_t z_0$. *The closed-loop system is*:

$$\dot{z}_t = A^{\mathrm{cl}}_t z_t; \quad t \geq 0. \tag{15}$$

Based on Theorem 1, the solvability of the coupled Riccati differential equations (12) is vital to the finite horizon Nash equilibrium solution. In the following, a necessary and sufficient condition is established for the solvability of the coupled Riccati differential equations.

Define

$$M = \begin{bmatrix} A & -S^1 & \cdots & -S^m \\ -Q_1 & -A^T & 0 & 0 \\ & \cdots & & \\ -Q^m & 0 & 0 & -A^T \end{bmatrix}$$

and

$$H_T = \begin{bmatrix} I_{(2nm)} & 0 & \cdots & 0 \end{bmatrix} e^{-MT} \begin{bmatrix} I_{(2nm)} \\ K^1_T \\ \vdots \\ K^m_T \end{bmatrix}.$$

It follows from Theorem 7.1 in Engwerda (2005) that the analytic solution of the closed-loop system is

$$z_t = \begin{bmatrix} I_{(2nm)} & 0 & \cdots & 0 \end{bmatrix} e^{-Mt} \begin{bmatrix} I_{(2nm)} \\ K^1_T \\ \vdots \\ K^m_T \end{bmatrix} H_T^{-1} z_0. \tag{16}$$

Theorem 7.1 in Engwerda (2005) provides an approach to judge if the solution exists for two-player games. This result can be generalised straightforwardly to m player games. Based on this theorem with m players, the formation control problem has the following result.

For a m-robot finite horizon formation control defined as a finite horizon open-loop Nash differential game by (11) and (12), the coupled Riccati differential equations (12) has a solution for every initial state z_0 on $[0, T]$ if and only if matrix H_T is invertible.

3.2. *Receding horizon Nash control*

To derive a state-feedback controller for practical uses in formation control, the open-loop Nash equilibrium solution can be combined with a receding horizon approach to synthesise a state-feedback controller: receding horizon Nash control. In this receding horizon Nash control, each robot computes its open-loop Nash equilibrium at each time instant, but is not committed to follow that equilibrium during the whole period. It only uses it to control one step. In the next step, this procedure repeats again.

Assuming the current time instant is t and the current state vector is z_t. At each time instant, the receding horizon control uses z_t as the initial state vector to find the finite horizon open-loop Nash equilibrium $\bar{u}_0$ based on the following cost function:

$$J_t^i = g_{t+T}^i + \int_t^{t+T} C_\tau^i(z_\tau, u_\tau)\mathrm{d}\tau. \qquad (17)$$

The receding horizon control signal is defined as:

$$u_t^i = \bar{u}_0^i = -R^{i^{-1}} B^{i^T} K_0^i z_t. \qquad (18)$$

As the control signal u_t^i depends on the current state z_t, the receding horizon Nash control is a state-feedback control. The existence conditions of the receding horizon Nash control is the same as those of the finite horizon open-loop Nash control, i.e. the receding horizon Nash control exists for every initial state z_0 if and only if matrix H_T is invertible.

The receding horizon Nash control needs to check whether or not the closed-loop system is stable. The closed-loop system with the receding horizon Nash control u_t^i (18) is

$$\dot{z}_t = \left(A - \sum_{i=1}^m S^i K_0^i\right) z_t = A_0^{\mathrm{cl}} z_t, \qquad (19)$$

where the closed-loop system matrix $A_0^{\mathrm{cl}} = A - \sum_{i=1}^m S^i K_0^i$. The following result can be made about the receding horizon Nash control.

The formation control defined as a finite horizon Nash differential game (11) and (17) has a receding horizon Nash control for every initial state z_0 if and only if matrix H_T is invertible. As long as all the eigenvalues of A_0^{cl} have negative real parts, the receding horizon Nash control is asymptotic stable.

3.3. *Distributed control*

The receding horizon Nash control signal (18) needs the state vector z_t, which includes all states from the formation team. However, the weight parameters ω^{ij} and μ^{ij} in the Nash game can be selected as 0 for robot i if robot j is not its neighbour. This selection will lead to the following matrix form of Q^i and K_T^i:

$$Q^i = \begin{bmatrix} q_{1,1}^i & \cdots & q_{1,j-1}^i & 0 & q_{1,j+1}^i & \cdots & q_{1,m}^i \\ & & & \cdots & & & \\ q_{j-1,1}^i & \cdots & q_{j-1,j-1}^i & 0 & q_{j-1,j+1}^i & \cdots & q_{j-1,m}^i \\ 0 & \cdots & 0 & 0 & 0 & \cdots & 0 \\ q_{j+1,1}^i & \cdots & q_{j+1,j-1}^i & 0 & q_{j+1,j+1}^i & \cdots & q_{j+1,m}^i \\ & & & \cdots & & & \\ q_{m,1}^i & \cdots & q_{m,j-1}^i & 0 & q_{m,j+1}^i & \cdots & q_{m,m}^i \end{bmatrix}$$

$$K_T^i = \begin{bmatrix} k_{1,1}^{i,T} & \cdots & k_{1,j-1}^{i,T} & 0 & k_{1,j+1}^{i,T} & \cdots & k_{1,m}^{i,T} \\ & & & \cdots & & & \\ k_{j-1,1}^{i,T} & \cdots & k_{j-1,j-1}^{i,T} & 0 & k_{j-1,j+1}^{i,T} & \cdots & k_{j-1,m}^{i,T} \\ 0 & \cdots & 0 & 0 & 0 & \cdots & 0 \\ k_{j+1,1}^{i,T} & \cdots & k_{j+1,j-1}^{i,T} & 0 & k_{j+1,j+1}^{i,T} & \cdots & k_{j+1,m}^{i,T} \\ & & & \cdots & & & \\ k_{m,1}^{i,T} & \cdots & k_{m,j-1}^{i,T} & 0 & k_{m,j+1}^{i,T} & \cdots & k_{m,m}^{i,T} \end{bmatrix}$$

where $q_{u,v}^i$ or $k_{u,v}^{i,T}$ is a block with size $(2n) \times (2n)$. Q^i and K_T^i has $m \times m$ blocks. The j-th block row or column consists of m zero blocks. It should be noted that matrix A has a block diagonal structure. Based on these matrix structures, it can be found that the j-th block row of solution K_t^i consists of m zero blocks from the coupled Riccati differential equations (12):

$$K_t^i = \begin{bmatrix} k_{1,1}^{i,t} & \cdots & k_{1,j-1}^{i,t} & k_{1,j}^{i,t} & k_{1,j+1}^{i,t} & \cdots & k_{1,m}^{i,t} \\ & & & \cdots & & & \\ k_{j-1,1}^{i,t} & \cdots & k_{j-1,j-1}^{i,t} & k_{j-1,j}^{i,t} & k_{j-1,j+1}^{i,t} & \cdots & k_{j-1,m}^{i,t} \\ 0 & \cdots & 0 & 0 & 0 & \cdots & 0 \\ k_{j+1,1}^{i,t} & \cdots & k_{j+1,j-1}^{i,t} & k_{j+1,j}^{i,t} & k_{j+1,j+1}^{i,t} & \cdots & k_{j+1,m}^{i,t} \\ & & & \cdots & & & \\ k_{m,1}^{i,t} & \cdots & k_{m,j-1}^{i,t} & k_{m,j}^{i,t} & k_{m,j+1}^{i,t} & \cdots & k_{m,m}^{i,t} \end{bmatrix}$$

Therefore, the receding horizon Nash control u_t^i (18) does not need the state z_t^j from non-neighbour robot j. If there are more than one robots in the team, which are not the neighbours of robot i, the same conclusion can be made. Thus, u_t^i is a distributed control law.

4. Formation control with network-induced delay

4.1. *Network-induced delay*

Most wireless networks can cause a random delay due to their narrow bandwidth. The stochastic behaviour is not desirable in robotic systems. Some effects are being made to obtain a deterministic communication behaviour in the networked embedded systems. Obtaining a constant network-induced delay when designing a network protocol is one way to obtain the deterministic communication behaviour (Roedig et al. 2006). In this article, we consider the communication network that has a constant delay for all robots.

Denote k as the sample step at time t and $t = k\delta$ ($k = 0, 1, 2, \ldots$), where δ is the sample period. For the delay $\tau = \delta - \lambda$ which is less than or equal to one sample period ($0 \le \lambda < \delta$), the system equations of robot i for the period $t \in [k\delta + \tau, (k+1)\delta + \tau)$ are expressed as:

$$\begin{aligned} \dot{z}_t^i &= az_t^i + bu_{t-\tau}^i \\ &= az_t^i + b\left(u_{t-\tau}^{i,i} + \sum_{(i,j)\in\mathcal{E}} u_{t-\tau}^{i,j}\right), \\ u_{t-\tau}^{i,i} &= -f^{i,i}z_{t-\tau}^i, u_{t-\tau}^{i,j} = -f^{i,j}z_{t-\tau}^j. \end{aligned} \tag{20}$$

where $[f^{i,1}, f^{i,2}, \ldots, f^{i,m}] = R^{i^{-1}}B^{i^T}K_0^i$ (see (18)). Collectively, we have

$$\begin{aligned} \dot{z}_t &= Az_t + Bu_{t-\tau}, \\ u_{t-\tau} &= -Fz_{t-\tau}, \end{aligned} \tag{21}$$

where

$$F = \begin{bmatrix} f^{1,1} & f^{1,2} & \ldots & f^{1,m} \\ f^{2,1} & f^{2,2} & \ldots & f^{2,m} \\ & & \vdots & \\ f^{m,1} & f^{m,2} & \ldots & f^{m,m} \end{bmatrix}.$$

By sampling the system (21) with period δ, we have (Franklin, Powell, and Workman 1997):

$$z_{k+1} = \Phi z_k + \Gamma_1 u_{k-1} + \Gamma_2 u_k, \tag{22}$$

where

$$\Phi = e^{A\delta} = I_{(m)} \otimes \begin{bmatrix} I_{(n)} & \delta I_{(n)} \\ 0 & I_{(n)} \end{bmatrix},$$

$$\Gamma_1 = \int_\lambda^\delta e^{A\eta}B\mathrm{d}\eta = I_{(m)} \otimes \begin{bmatrix} \frac{1}{2}(\delta^2 - \lambda^2)I_{(n)} \\ (\delta - \lambda)I_{(n)} \end{bmatrix},$$

$$\Gamma_2 = \int_0^\lambda e^{A\eta}B\mathrm{d}\eta = I_{(m)} \otimes \begin{bmatrix} \frac{1}{2}\lambda^2 I_{(n)} \\ \lambda I_{(n)} \end{bmatrix}.$$

By defining $x_k = [z_k^T, u_{k-1}^T]^T$ as the augmented state vector, the augmented closed-loop system is

$$x_{k+1} = \tilde{\Phi}x_k, \tag{23}$$

where

$$\tilde{\Phi} = \begin{bmatrix} \Phi - \Gamma_2 F & \Gamma_1 \\ -F & 0 \end{bmatrix}.$$

The closed-loop system is stable if and only if all eigenvalues of the matrix $\tilde{\Phi}$ are within the unit circle.

If the delay is longer than one sampling period δ, the delay can be denoted as $\tau = h\delta - \lambda$, where $0 \le \lambda < \delta$ and $h > 1$. The system state equation is

$$\begin{aligned} \dot{z}_t^i &= az_t^i + bu_t^i \\ &= az_t^i + b(u_t^{i,i} + \sum_{(i,j)\in\mathcal{E}} u_{t-\tau}^{i,j}), \\ u_t^{i,i} &= -f^{i,i}z_t^i, u_{t-\tau}^{i,j} = -f^{i,j}z_{t-\tau}^j. \end{aligned} \tag{24}$$

Collectively, we have

$$\begin{aligned} \dot{z}_t &= Az_t + B(u_t'' + u_{t-\tau}'), \\ u_t'' &= -F_1 z_t, u_{t-\tau}' = -F_2 z_{t-\tau}, \end{aligned} \tag{25}$$

where

$$u_t'' = \begin{bmatrix} u_t^{1,1} \\ u_t^{2,2} \\ \vdots \\ u_t^{m,m} \end{bmatrix}, \quad u_{t-\tau}' = \begin{bmatrix} \sum_{(1,j)\in\mathcal{E}} u_{t-\tau}^{1,j} \\ \sum_{(2,j)\in\mathcal{E}} u_{t-\tau}^{2,j} \\ \vdots \\ \sum_{(m,j)\in\mathcal{E}} u_{t-\tau}^{m,j} \end{bmatrix},$$

$$F_1 = \begin{bmatrix} f_1^1 & 0 & \ldots & 0 \\ 0 & f_2^2 & \ldots & 0 \\ & & \vdots & \\ 0 & 0 & \ldots & f_m^m \end{bmatrix}, \quad F_2 = \begin{bmatrix} 0 & f_2^1 & \ldots & f_m^1 \\ f_1^2 & 0 & \ldots & f_m^2 \\ & & \vdots & \\ f_1^m & f_2^m & \ldots & 0 \end{bmatrix}.$$

The discrete system state equation is:

$$\begin{aligned} z_{k+1} &= \Phi z_k + \Gamma u_k'' + \Gamma_1 u_{k-h}' + \Gamma_2 u_{k-h+1}' \\ &= (\Phi - \Gamma F_1)z_k + \Gamma_1 u_{k-h}' + \Gamma_2 u_{k-h+1}' \\ &= \Phi_1 z_k + \Gamma_1 u_{k-h}' + \Gamma_2 u_{k-h+1}', \end{aligned} \tag{26}$$

where $\Phi_1 = \Phi - \Gamma F_1$ and

$$\Gamma = \int_0^\delta e^{A\eta}B\mathrm{d}\eta = I_{(m)} \otimes \begin{bmatrix} \frac{1}{2}\delta^2 I_{(n)} \\ \delta I_{(n)} \end{bmatrix}.$$

By defining $x_k = [z_k^T, u'^T_{k-h}, \ldots, u'^T_{k-1}]^T$ as the augmented state vector, the augmented closed-loop system is

$$x_{k+1} = \tilde{\Phi}x_k, \tag{27}$$

where

$$\tilde{\Phi} = \begin{bmatrix} \Phi_1 & \Gamma_1 & \Gamma_2 & \cdots & 0 \\ 0 & 0 & I_{(nm)} & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \cdots & I_{(nm)} \\ -F_2 & 0 & 0 & \cdots & 0 \end{bmatrix}.$$

The closed-loop system is stable if and only if all eigenvalues of the matrix $\tilde{\Phi}$ are within the unit circle.

4.2. *Compensation for network-induced delay*

The compensation for the delay will improve the control performance. We begin with the collective formation system to make the compensation based on state prediction and go down to the individual robot to check if the distributed feature is still maintained under the compensation.

Assume the delay is integer number of the sample periods, $h\delta$ $(h=1,2,\ldots)$. At step k, the system obtains a delayed state z_{k-h}. To compensate for the delay h, the system needs to estimate its current state based on the delayed state z_{k-h}. The discrete state equation is

$$z_{k+1} = \Phi z_k + \Gamma u_k. \tag{28}$$

The estimated current state at k is denoted as $\hat{z}_{k|k-h}$. The controller with the delay compensation is defined as

$$u_k = -F_1 z_k - F_2 \hat{z}_{k|k-h}. \tag{29}$$

To estimate $\hat{z}_{k|k-h}$ starting from step $k-h$, a simple estimator can be used

$$\begin{aligned} \hat{z}_{k-h+1|k-h} &= \Phi \hat{z}_{k-h|k-h-1} + \Gamma u_{k-h} + G(z_{k-h} - \hat{z}_{k-h|k-h-1}) \\ &= G z_{k-h} + (\Phi - G)\hat{z}_{k-h|k-h-1} + \Gamma u_{k-h}. \end{aligned} \tag{30}$$

For the steps from $k-h+1$ up to k, the estimated state $\hat{z}_{k-h+1|t-h}$ in (30) can be used as the initial state. But the individual robot does not know the neighbour's control signals from $k-h+1$ up to k due to the delay. Therefore, the approximation has to be made to the estimation. In the following approximated estimation, the control signal u_{k-h} at step $k-h$ is used for all steps from $k-h+1$ up to k.

$$\begin{aligned} \hat{z}_{k-h+2|k-h} &= \Phi \hat{z}_{k-h+1|k-h} + \Gamma u_{k-h} \\ &\vdots \\ \hat{z}_{k|k-h} &= \Phi \hat{z}_{k-1|k-h} + \Gamma u_{k-h}. \end{aligned}$$

Finally, the estimated current state is:

$$\begin{aligned} \hat{z}_{k|k-h} &= \Phi^{h-1} G z_{k-h} + \Phi^{h-1}(\Phi - G)\hat{z}_{k-h|k-h-1} \\ &\quad + \sum_{j=1}^{h} \Phi^{h-j} \Gamma u_{k-h} \\ &= \Theta_1 z_{k-h} + \Theta_2 \hat{z}_{k-h|k-h-1} + \Theta_3 u_{k-h}, \end{aligned} \tag{31}$$

where $\Theta_1 = \Phi^{h-1} G$, $\Theta_2 = \Phi^{h-1}(\Phi - G)$ and $\Theta_3 = \sum_{j=1}^{h} \Phi^{h-j}\Gamma$.

From (29), the control signal is obtained as

$$\begin{aligned} u_k &= -F_1 z_k - F_2 \hat{z}_{k|k-h} \\ &= -F_1 z_k - F_2(\Theta_1 z_{k-h} + \Theta_2 \hat{z}_{k-h|k-h-1} + \Theta_3 u_{k-h}). \end{aligned} \tag{32}$$

From the system state Equation (28), the next state is obtained as:

$$\begin{aligned} z_{k+1} &= \Phi z_k + \Gamma u_k \\ &= \Phi_1 z_k - \Gamma F_2(\Theta_1 z_{k-h} + \Theta_2 \hat{z}_{k-h|k-h-1} + \Theta_3 u_{k-h}). \end{aligned} \tag{33}$$

By defining an augmented state vector $X_k = [z_k^T, \ldots, z_{k-h+1}^T, z_{k-h}^T, \hat{z}_{k-h|k-h-1}^T, u'^T_{k-1}, \ldots, u'^T_{k-h}]^T$, the augmented state equation is written as:

$$X_{k+1} = \Lambda X_k, \tag{34}$$

where

$$\Lambda = \begin{bmatrix} \Phi_1 & \ldots & 0 & -\Gamma F_2 \Theta_1 & -\Gamma F_2 \Theta_2 & 0 & \ldots & 0 & -\Gamma F_2 \Theta_3 \\ I & \ldots & 0 & 0 & 0 & 0 & \ldots & 0 & 0 \\ & & & & \vdots & & & & \\ 0 & \ldots & I & 0 & 0 & 0 & \ldots & 0 & 0 \\ 0 & \ldots & 0 & G & \Phi - G & 0 & \ldots & 0 & \Gamma \\ 0 & \ldots & 0 & -F_2\Theta_1 & -F_2\Theta_2 & 0 & \ldots & 0 & -F_2\Theta_3 \\ 0 & \ldots & 0 & 0 & 0 & I & \ldots & 0 & 0 \\ & & & & \vdots & & & & \\ 0 & \ldots & 0 & 0 & 0 & 0 & \ldots & I & 0 \end{bmatrix}.$$

So we can say that the closed-loop system is stable if and only if all eigenvalues of the matrix Λ are within the unit circle.

Now, we need to check if the controller (32) is a distributed one as in the case without delay. As Φ, G and Γ are block diagonal matrices and each block is related to one robot, Θ_1, Θ_2 and Θ_3 defined in (31) also have such property. So we can see from (29) that the controller (32) is still a distributed controller. However, it needs more information from its neighbours, including z^j_{k-h}, u^j_{k-h} and $\hat{z}^j_{k-h|k-h-1}$. As all information from a neighbour j can be integrated $\hat{z}^j_{k|k-h}$ by using (31), robot j only needs to send out its own estimated current state $\hat{z}^j_{k|k-h}$. Actually robot j sends out its estimated current state $\hat{z}^j_{k+h|k}$ at step k. After delay, robot i receives the delayed state $\hat{z}^j_{k|k-h}$. To estimate

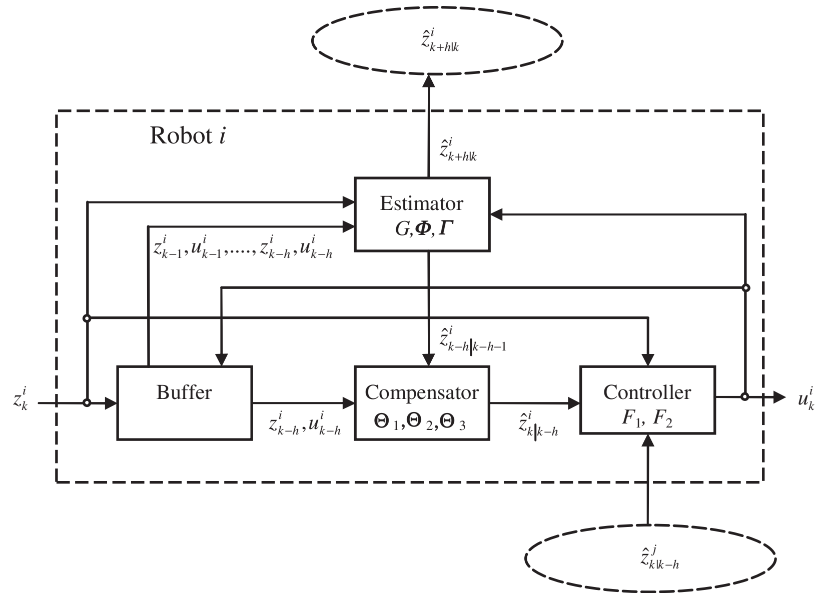

Figure 1. Distributed controller.

$\hat{z}_{k+h|k}$ for each robot, Equation (31) can be used at step k in the following form:

$$\hat{z}_{k+h|k} = \Theta_1 z_k + \Theta_2 \hat{z}_{k|k-1} + \Theta_3 u_k. \tag{35}$$

Each robot sends $\hat{z}^i_{k+h|k}$ to its neighbours. And it also receives the delayed state $\hat{z}^j_{k|k-h}$ from neighbour j. Inside each robot, the controller has several necessary components and is shown in Figure 1 where only one neighbour robot j is considered. In the diagram, robot i obtains its own state z^i_k through its own sensors and the delayed state $\hat{z}^j_{k|k-h}$ through communication. They are the inputs of the control algorithm. Robot i generates the control signal u^i_k as the output of the controller and sends estimated state $z^i_{k+h|k}$ to neighbours. The buffer is used to keep its current state and control signal for $h-1$ steps. The estimator uses the parameters G, Φ and Γ, buffered states and control signals, and Equation (30) to estimate $\hat{z}^i_{k-h+1|k-h}, \ldots, \hat{z}^i_{k|k-1}$. It also generates the estimated current state $\hat{z}^i_{k+h|k}$. The compensator uses the parameters Θ_1, Θ_2, Θ_3 and (31) to obtain the estimate current state $\hat{z}^i_{k|k-h}$. The controller uses the control gains (F_1, F_2) and (32) to generate the control signal u^i_k.

The computational complexity depends on the number of neighbours. Let N^i denote the number of neighbours of robot i. The computational complexity of the controller is about $(2n+2n)N^i = 4nN^i$. The computational complexity of the estimator is about $(2n+2n+n)h = 5nh$. The computational complexity of the compensator is about $2n+2n+n = 5n$. In total, the computational complexity is about $4nN^i + 5n(h+1)$.

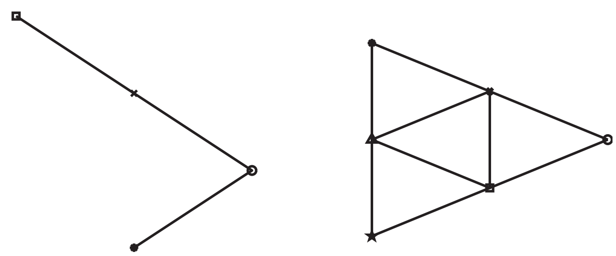

Figure 2. Triangle formation shapes.

5. Simulations

The dimension of the coordinate vectors for all robots is $n=2$. Simulation tests a triangle formation shape with four robots ($m=4$) and with six robots ($m=6$) (Figure 2). The vertex nodes 1, 2, 3, 4, 5 and 6 are represented by circle, cross, square, star, triangle and pentagram in the figure. The tracking trajectories are assumed to be a circle and a sinusoid. The circle is defined $q^r_t = [\cos(t), \sin(t)]^T$; $t \geq 0$. The sinusoid is defined as $q^r_t = [t, \sin(t)]^T$; $t \geq 0$.

5.1. *Distributed control*

The distributed control is validated via testing the triangle formation. The following two tasks are simulated. The difference between two tasks is whether or not the leader robot 1 uses the formation cost.

- T1: The leader robot 1 uses the cost function (10), which includes a formation cost represented by K^1_T and Q^1, and a tracking cost represented by k^1_T and q^1. The follower robots

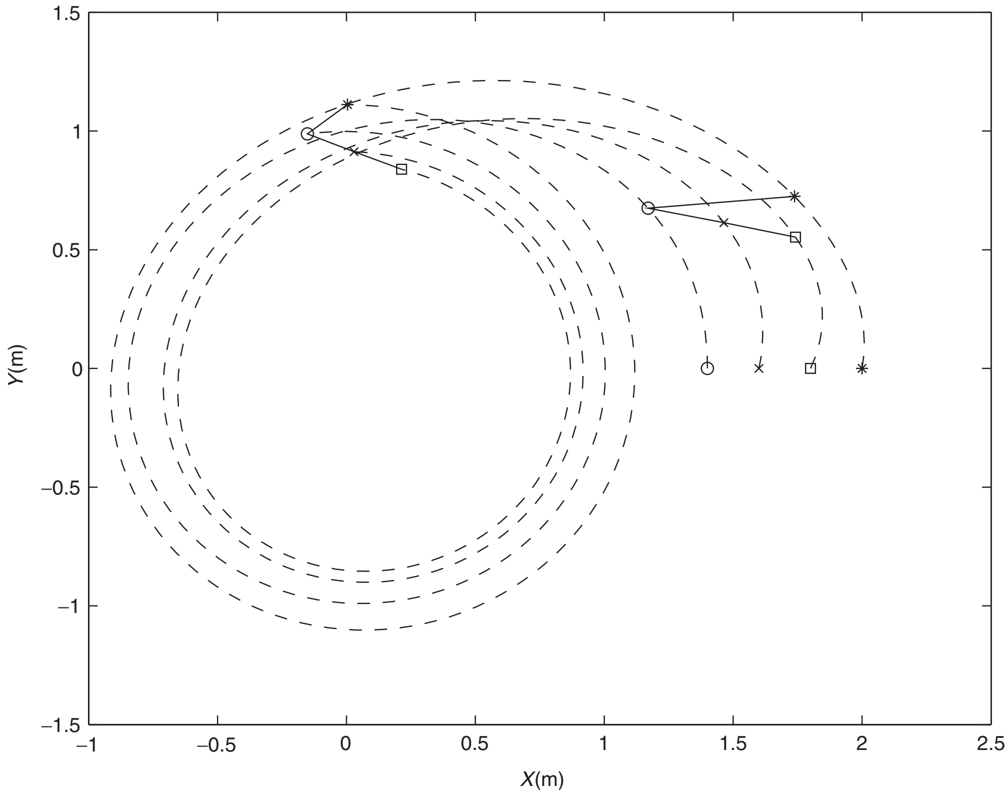

Figure 3. Triangle formation tracking a circle in T1.

use the cost function (9), which only includes a formation cost represented by K_T^i and Q^i, $i=2, 3, 4$.

- T2: The leader robot 1 only uses the tracking cost function represented by k_T^1 and q^1 (without the formation cost, $K_t^1 = 0$ and $Q^1 = 0$ in (10)). The follower robots use the same cost function (9).

The leader robot's tracking weight matrix is selected as $k_T^1 = q^1 = 5I_{(2n)}$. In the triangle formation, the neighbours of the leader robot 1 are robot 2 and 4, the neighbours of robot 2 are robot 1 and 3, the neighbour of robot 3 is robot 2 and the neighbour of robot 4 is robot 1. So, their formation cost weight matrixes are selected as:

$$W^1 = K_T^1 = \begin{bmatrix} 5 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 5 \end{bmatrix}, \quad W^2 = W_T^2 = \begin{bmatrix} 5 & 0 & 0 \\ 0 & 5 & 0 \\ 0 & 0 & 0 \end{bmatrix},$$
$$W^3 = W_T^3 = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 5 & 0 \\ 0 & 0 & 0 \end{bmatrix}, \quad W^4 = W_T^4 = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 5 \end{bmatrix}.$$

The cost weight matrix for control signal is selected to be $R^i = [\begin{smallmatrix} 1 & 0 \\ 0 & 1 \end{smallmatrix}]$ for $i=1, 2, 3, 4$. Solutions to the coupled Riccati differential equations of the finite horizon open-loop Nash differential games can be found by using terminal values and the backward iteration. The finite horizon length $T=10$ s and sample time is $\delta=0.1$ s. Through calculation, it can be seen that $H(10)$ is invertible and all eigenvalues of $A_{\mathrm{cl}}(0)$ have negative real parts. Therefore, the receding horizon Nash control is asymptotic stable.

The trajectories of the four robots in T1 are shown in Figure 3. The leader robot 1 uses both tracking cost function and formation cost function. The results show that all four trajectories converge to a triangle shape during the circle tracking. All control error signals converge to zero, i.e. the robot control signals converge to the tracking control signals. x and y position errors between robots and their tracking trajectories are shown in Figure 4. It can be seen that the position errors converge to zero, i.e. all robots finally move in the circle trajectories.

In T2, the leader robot 1 only uses the tracking cost function. Again all robots can track the circle after the initial transient process. The position errors are shown in Figure 5. All these errors have the convergent property. Due to the only use of the tracking cost function, the leader robot just tracks the circle trajectory without considering the formation error. It can be seen that its y position errors

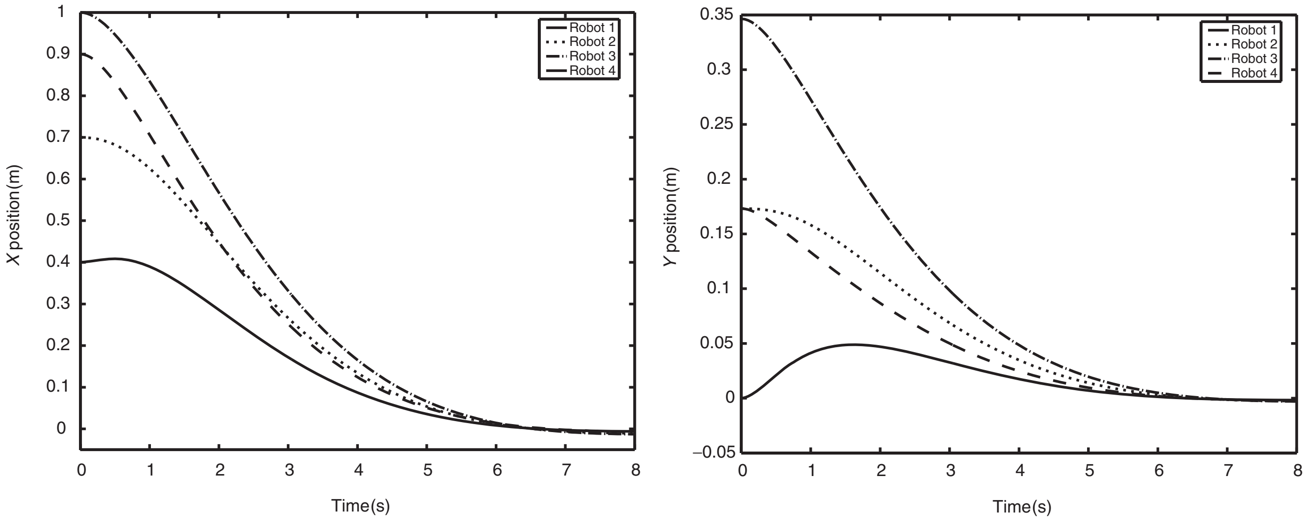

Figure 4. x (left) and y (right) position errors in triangle formation in T1.

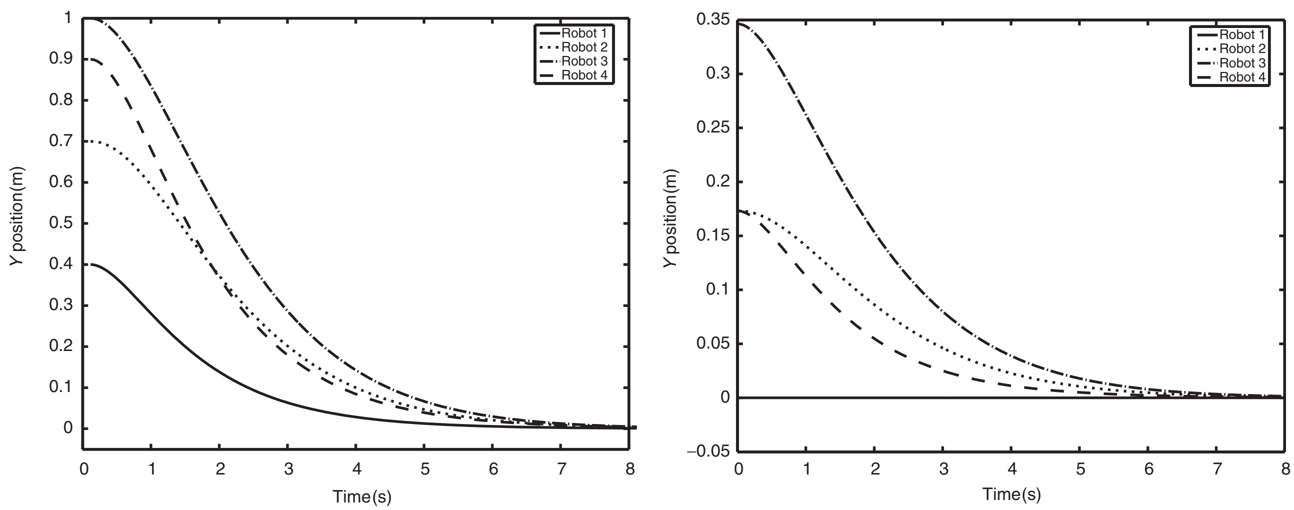

Figure 5. x (left) and y (right) position errors in triangle formation in T2.

(solid line) are kept as zero. This is due to the initial y position being the same as the tracking trajectory. The leader robot only adjusts its x position error to catch up with the circle. The use of the leader without a cost on the formation can make the formation control work in a leader–follower structure. The leader robot 1 does not need state information from the followers and it can track the desired trajectory better.

5.2. *Distributed control with network-induced delay*

To justify the proposed distributed formation control with compensation for delay, three simulations are made for both formation patterns. The first one is the simulation without delay. The second one is the simulation with delay and the third one is the simulation with compensation. All parameters are the same in three simulations. The finite horizon length is $T = 30\,\mathrm{s}$ and it can guarantee the distributed control without delay is stable by checking that $H(T)$ is invertible and the eigenvalues of $A_{cl}(0)$ have negative real parts. The number of delay step is $h = 5$. The estimator gain is $G = I_{2nm}$. The cost weight parameters are $R^i = 50I_n$ and $W^i = W^i_T = I_m$. For the formation with six robots, the formation graph has six nodes and nine edges. Based on all above values, it can be checked by using MATLAB that the eigenvalues of $\tilde{\Phi}$ and Λ are within the unit circle, so the distributed formation control systems with delay and compensation are stable.

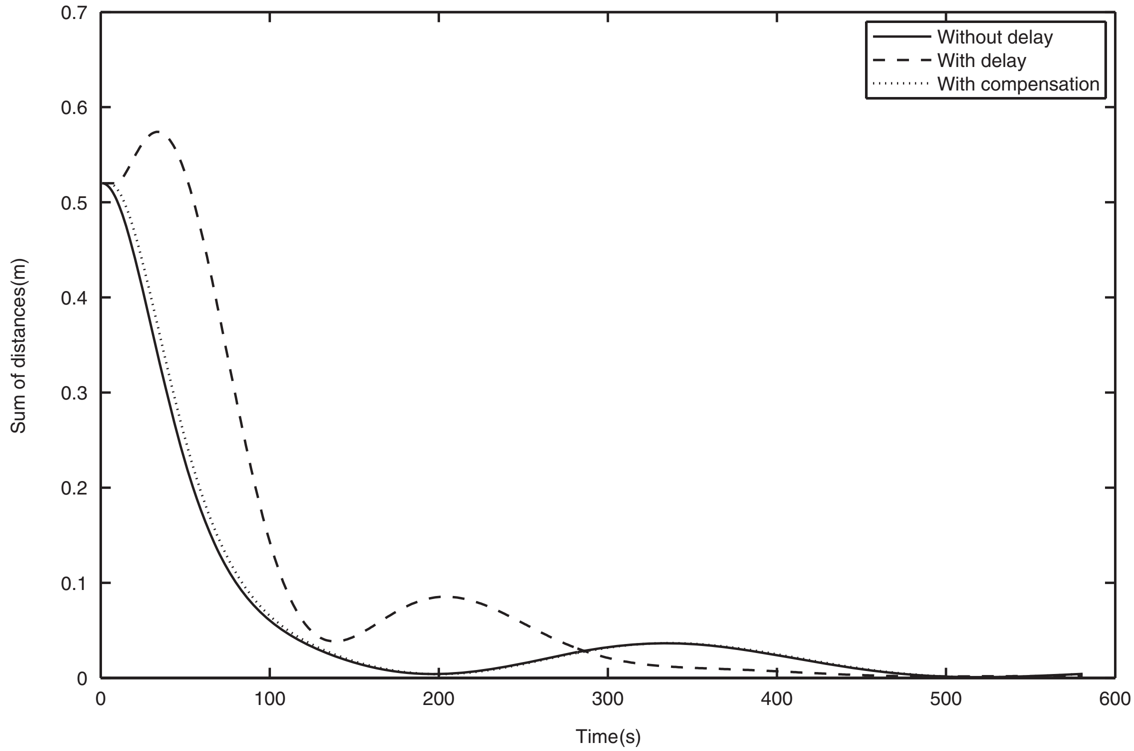

Figure 6. Four robots track a circle.

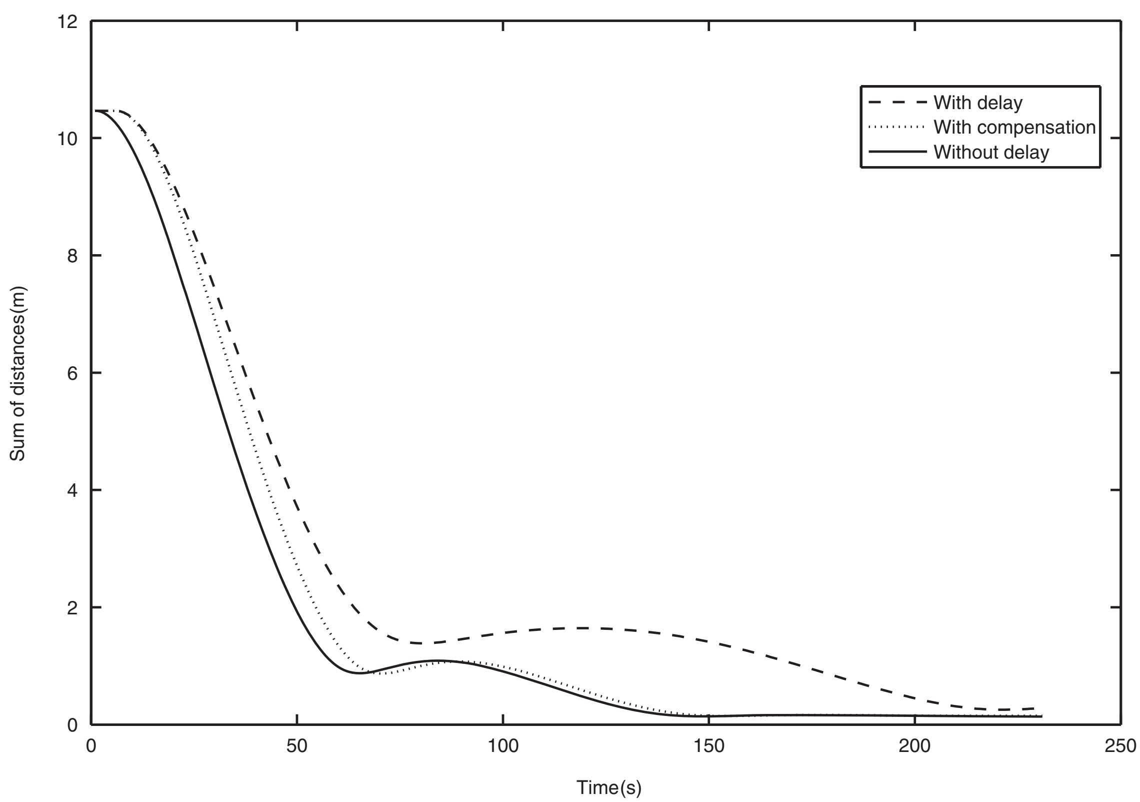

Figure 7. Six robots track a sinusoid.

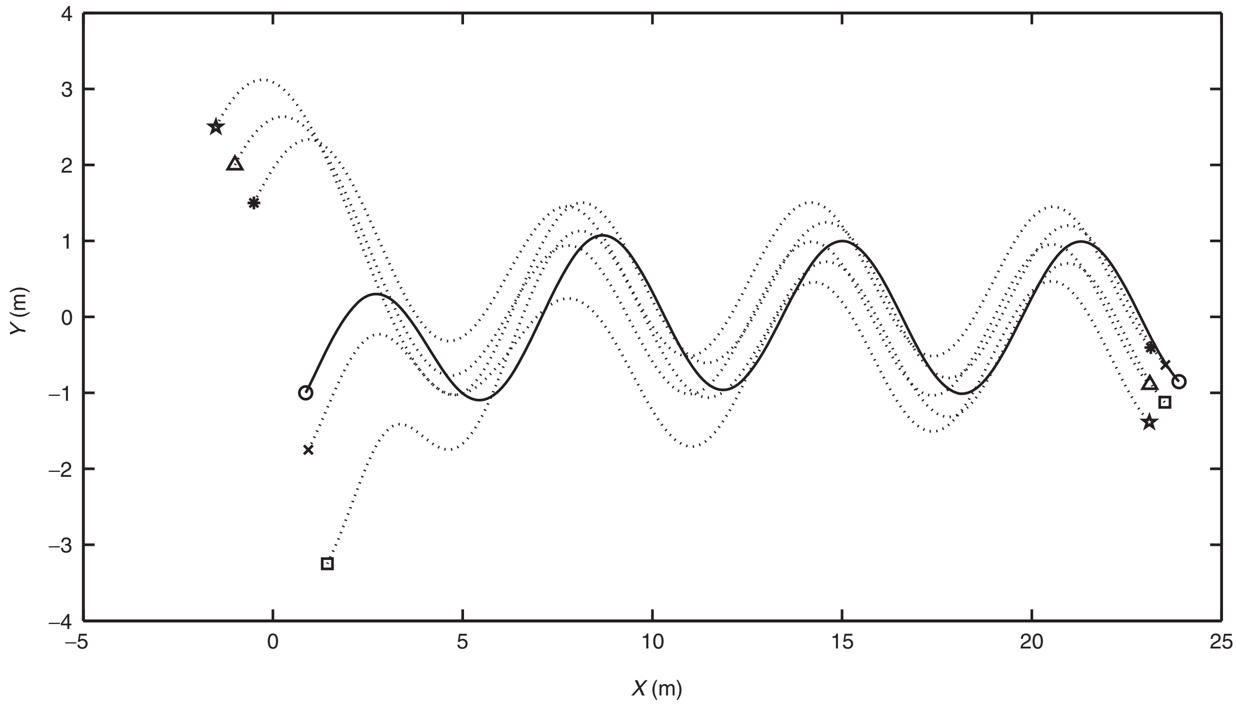

Figure 8. Six robots track a sinusoid without delay.

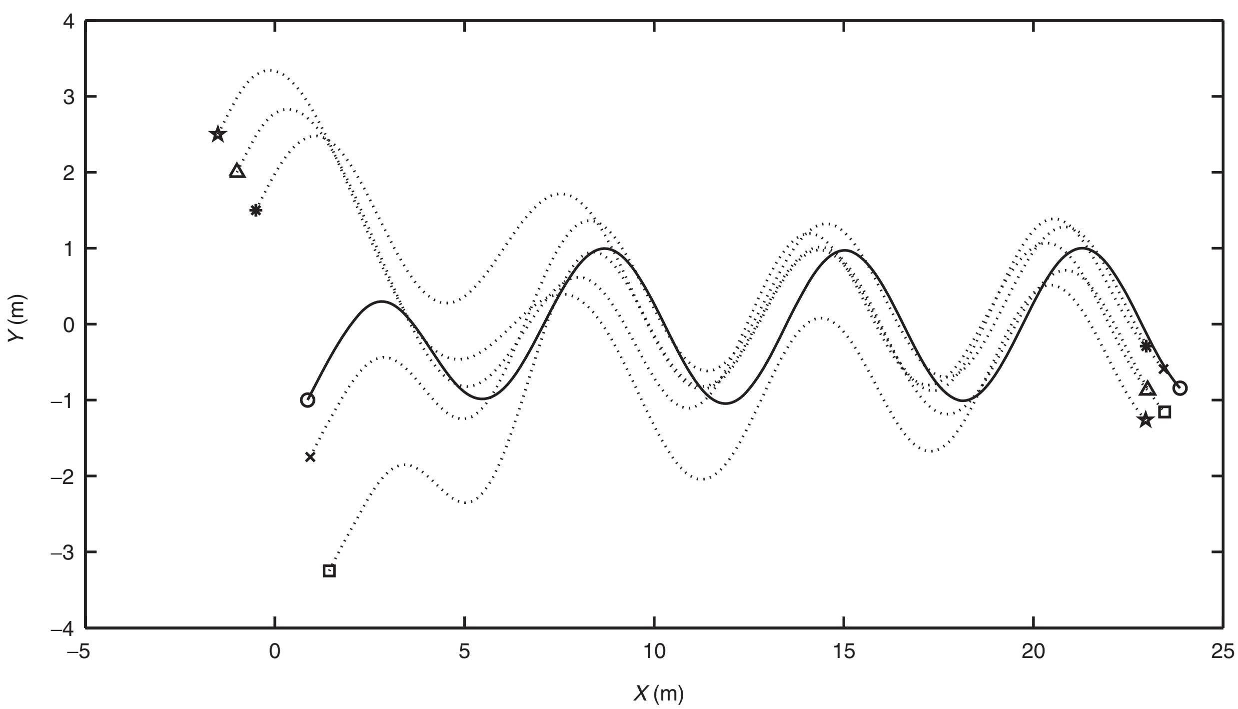

Figure 9. Six robots track a sinusoid with delay.

The main simulation results are shown in Figure 6 where four robots track a circle and Figure 7 where six robots track a sinusoid. The figures show the sum of all distance errors between neighbour robots. It can be seen from both figures that the sum of distance errors in all the three simulations are convergent. With a closer look, it is found that at the beginning of the tracking, the simulation result with delay is very close

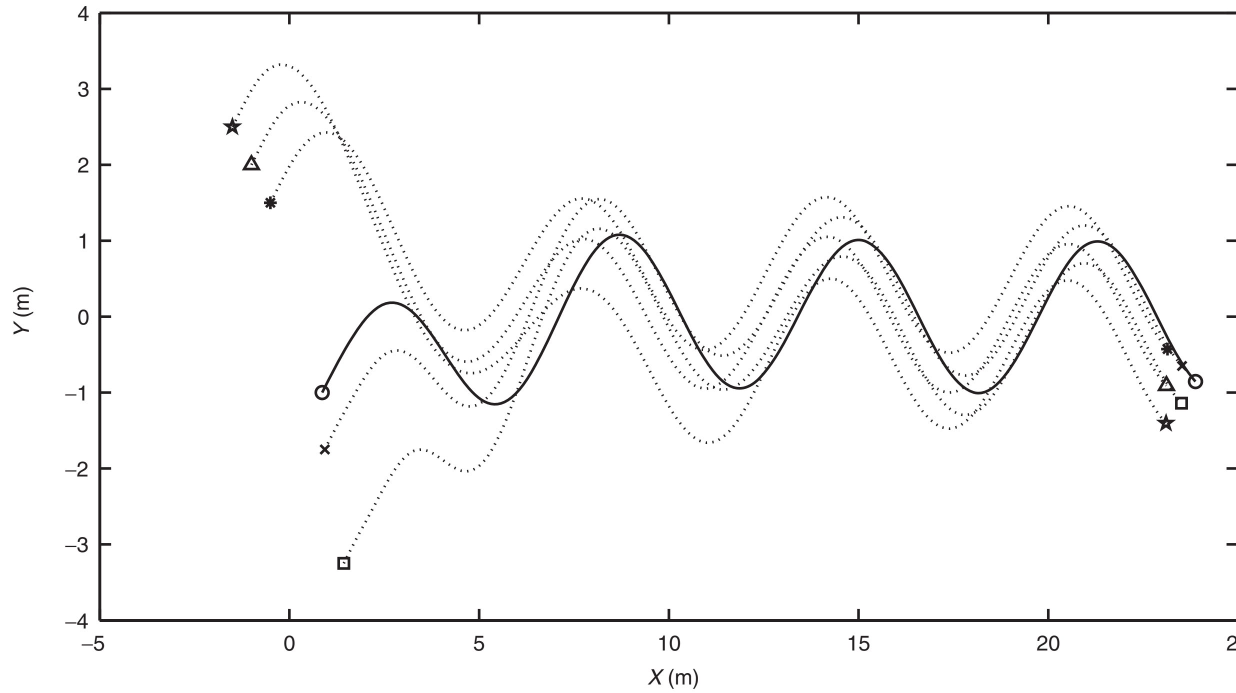

Figure 10. Six robots track a sinusoid with compensation.

to that with compensation. This just lasts for a very short period. After this initial period, the simulation result with compensation quickly moves away from the result with delay and moves towards the result without delay. At the end, the simulation result with compensation is very close to that without delay, while the result with delay is far away from them. It demonstrates that although the system without delay is stable, the formation error tends to fluctuate more than the system without delay. This is due to the fact that the induced delay leads to the result that the formation controller cannot obtain the current state information, it uses old state information to control the robots, eventually it generates large formation error. However, the compensation made for the delay can predict the current state information based on the delayed system model and therefore the compensated formation controller generates better results than that without compensation.

The tracking trajectories of the simulations for without delay, with delay and with compensation are shown in Figures 8, 9, and 10. The trajectory of the simulation without delay has a similar behaviour as that with compensation, especially at the beginning (between 0 to 5 m in x-axis), while the trajectory of the simulation with delay experiences longer time to converge to the sinusoid. At the end of tracking, the simulations without delay and with compensation end up with a proper triangle shape. By looking carefully at the position of the robot marked with 'star', it can be seen that the triangle shape at the end of tracking is not properly formed in Figure 10. Long tracking time is required for the simulation with delay to establish a required triangle.

6. Conclusions

This article investigates two issues in the formation control problem. One is the implementation of distributed control and another is the stability analysis of formation control systems with network-induced delay. For the first problem, we use non-cooperative differential games to establish the distributed formation control. By building the Nash equilibria, all team members can find a self-enforcing controller. The state-feedback control can be synthesised through the receding horizon Nash control.

For the second problem, the effect of network-induced constant delay on the distributed formation control is analysed and the compensation for the delay is made. An augmented linear closed-loop system is built to establish the stability condition.

In the next step, we will consider the application of proposed distributed control to real robots. The first step is to build up a wireless *ad hoc* network for communication. The protocol for constant delay will be designed. Alternatively, the analysis for network-induced random delay and packet dropouts can be made.

References

Abou-Kandil, H., Freiling, G., Ionescu, V., and Jank, G. (2003), *Matrix Riccati Equations*, Switzerland: Birkhauser Verlag.

Balch, T., and Arkin, R. (1998), 'Behavior-based Formation Control for Multirobot Systems,' *IEEE Transactions on Robotics and Automation*, 14(2), 926–939.

Basar, T., and Olsder, G. (1995), *Dynamic Noncooperative Game Theory*, New York: SIAM.

Camponogara, E., Jia, D., Krogh, B., and Talukdar, S. (2002), 'Distributed Model Predictive Control,' *IEEE Control Systems Magazine*, 22(1), 44–52.

Das, A., Fierro, R., Kumar, V., Ostrowski, J., Spletzer, J., and Taylor, C. (2002), 'A Vision-based Formation Control Framework,' *IEEE Transactions on Robotics and Automation*, 18(5), 813–825.

Desai, J.P., Ostrowski, J.P., and Kumar, V. (2001), 'Modelling and Control of Formations of Nonholonomic Mobile Robots,' *IEEE Transactions on Robotics and Automation*, 17(6), 905–908.

Dunbar, W., and Murray, R. (2006), 'Model Predictive Control of Coordinated Multi-vehicle Formations,' *Automatica*, 2(4), 549–558.

Engwerda, J.C. (2005), *LQ Dynamic Optimization and Differential Games*, London: John Wiley & Sons Ltd.

Fierro, R., Das, A., Kumar, V., and Ostrowski, J.P. (2001), 'Hybrid Control of Formation of Robots,' in *Proceedings of the IEEE International Conference of Robotics and Automation*, 21–26 May 2001, Seoul, Korea, pp. 157–162.

Franklin, G.F., Powell, J.D., and Workman, M. (1997), *Digital Control of Dynamic Systems*, California: Addison Wesley–Longman Inc.

Giulietti, F., Pollini, L., and Innocenti, M. (2000), 'Autonomous Formation Flight,' *IEEE Control System Magazine*, 20(6), 34–44.

Godsil, C., and Royle, G. (2001), *Algebraic Graph Theory*, New York: Springer-Verlag.

Liu, G.P., Xia, Y.Q., Chen, J., Rees, D., and Hu, W. (2007), 'Networked Predictive Control of Systems with Random Network Delays in Both Forward and Feedback Channels,' *IEEE Transactions on Industrial Electronics*, 54(3), 1282–1297.

McInnes, C.R. (1995), 'Autonomous Ring Formation for a Planar Constellation of Satellites,' *AIAA Journal of Guidance, Control and Dynamics*, 18(5), 1215–1217.

Nesic, D., and Teel, A.R. (2004), 'Input–output Stability Properties of Networked Control Systems,' *IEEE Transactions on Automatic Control*, 49(10), 1650–1667.

Nilsson, J. (1998), 'Real-time Control Systems with Delays,' Ph.D. dissertation, Lund Institute of Technology, Lund, Sweden.

Ogren, P., Egerstedt, M., and Hu, X. (2002), 'A Control Lyapunov Function Approach to Multiagent Coordination,' *IEEE Transactions on Robotics and Automation*, 18(5), 847–851.

Pant, A., Seiler, P., and Hedrick, K. (2002), 'Mesh Stability of Look-ahead Interconnected Systems,' *IEEE Transactions on Automatic Control*, 47(2), 403–407.

Park, H.S., Kim, Y.H., Kim, D.S., and Kwon, W.H. (2002), 'A Scheduling Method for Network Based Control Systems,' *IEEE Transactions on Control Systems Technology*, 10(3), 318–330.

Raffard, R.L., Tomlin, C.J., and Boyd, S.P. (2004), 'Distributed Optimisation for Cooperative Agents: Application to Formation Flight,' in *Proceedings of the 43rd IEEE Conference on Decision and Control*, Atlantis, Paradise Island, Bahamas, pp. 2453–2459, December.

Roedig, U., Barroso, A., and Sreenan, C.J. (2006), '*f*-mac: A Deterministic Media Access Control Protocol Without Time Synchronization,' in *Proceedings of the third IEEE European Workshop on Wireless Sensor Networks (EWSN2006)*, Zurich, Switzerland: Springer, pp. 47–50.

Stipanovic, D.M., Inalhan, G., Teo, R., and Tomlin, C. (2004), 'Decentralised Overlapping Control of a Formation of Unmanned Aerial Vehicles,' *Automatica*, 40(8), 1285–1296.

Swaroop, D., and Hedrick, J.K. (1996), 'String Stability of Interconnected Systems,' *IEEE Transactions on Automatic Control*, 41(3), 349–357.

Tanner, H.G., Loizou, S.G., and Kyriakopoulos, K.J. (2003), 'Nonholonomic Navigation and Control of Multiple Mobile Manipulators,' *IEEE Transactions on Robotics and Automation*, 19(1), 53–64.

Walsh, G.C., Beldiman, O., and Bushnell, L.G. (2001), 'Asymptotic Behaviour of Non-linear Networked Control Systems,' *IEEE Transactions on Automactic Control*, 46(7), 1093–1097.

Weimerskirch, H., Martin, J., Clerquin, Y., Alexandre, P., and Jiraskova, S. (2001), 'Energy Saving in Flight Formation,' *Nature*, 413, 697–698.

Zhang, W., Brannicky, M.S., and Philips, S.M. (2001), 'Stability of Networked Control Systems,' *IEEE Control Systems Magazine*, 21(1), 84–99.