TÜBİTAK

Turkish Journal of Electrical Engineering & Computer Sciences

http://journals.tubitak.gov.tr/elektrik/

Research Article

Turk J Elec Eng & Comp Sci
(2013) 21: 1182 – 1198
© TÜBİTAK
doi:10.3906/elk-1111-46

Shape control of cyclic networks in multirobot formations

Tolga EREN[*]
Department of Electrical and Electronics Engineering, Kırıkkale University, Kırıkkale, Turkey

Received: 19.11.2011 • **Accepted:** 09.02.2012 • **Published Online:** 03.06.2013 • **Printed:** 24.06.2013

Abstract: The first part of this paper examines cycles at the network level of sensing and control architectures that are needed to maintain the shape of a multiagent formation in 2-dimensional space, while the formation moves as a cohesive whole. The key tools used in the paper are rigidity theory and graph theory. The second part of the paper focuses on the planner level and controller level of the architecture by designing the fuzzy planner and fuzzy controller of individual agents. This permits the decomposition of the complex coordination problem into a series of smaller ones. The fuzzy planner and the fuzzy controller on kinematic unicycles are simulated for an exemplary cyclic formation generated by the results obtained in the first half of the paper. The validity of the proposed approach in the shape control of multiagent formations is verified through simulation experiments.

Key words: Formation control, networked control, cooperative control, multirobot formations, autonomous agents, rigidity theory, rigid graph theory, fuzzy control, fuzzy systems

1. Introduction

1.1. Shape and rigidity

Formation control of multiagent systems consisting of robots or unmanned vehicles with sensors and actuators has been an active research topic [1, 2, 3]. A fundamental task for autonomous agent formations is formation shape control. Obviously, if the distance between any pair of agents is kept constant, the shape of the formation is maintained. In this paper, attention is confined to maintaining the formation shape by explicitly keeping "some" interagent distances constant. Rigidity theory has become increasingly popular in the research on this topic. In this formalism, agents are represented by vertices in a graph. The edges in the graph represent coordination, sensing, and communication. A formation corresponding to a rigid graph benefits from its local unique realization. Eren et al. [4] provided a theoretical foundation for shape maintenance in terms of rigid graph theory. It was shown that an autonomous formation has local uniqueness if and only if its underlying graph is rigid. Since then, several authors have applied rigid graph theory to the control of autonomous formations. For example, in a series of papers [5, 6, 7, 8, 9] and the references therein, the authors showed the relevance and effectiveness of the notions of rigidity in formation control. Rigid graph theory concepts have also found a natural application in sensor network localization and robot network localization [10, 11, 12, 13].

1.2. Leader–follower architecture and persistence

The execution of the task of maintaining a prescribed distance between a pair of agents could be the responsibility of both agents. A disadvantage of the bidirectional control is that it relies on 2-way communication: if the control

[*]Correspondence: erente@gmail.com

of agent A uses the position of agent B, then the control of agent B uses the position of agent A.

Alternatively, the execution of the task could be the responsibility of one nominated agent of the pair. In this unidirectional case, only 1 of the 2 agents has to sense the position of the other agent or receive the position information broadcast by the other agent and make decisions on its own. This unilateral responsibility is captured by assigning a direction to the relevant edge in the graph, resulting in a digraph. A directed edge from vertex u to vertex v appears when agent v has the task of maintaining a specified distance from agent u, and agent u is unconstrained in its own motions with respect to the motion of v. The network topology of such a formation is called leader–follower architecture [14]. (Here, a similar direction convention is adopted as in communication/computer networks, i.e. a link is directed from leader to follower representing information flow as it is directed from source to sink. Some cited research papers adopt the direction convention the other way around. The presentation and notations are modified to be consistent with the convention adopted in this paper.) For formations with bilateral responsibility, rigidity theory suffices to create formations that maintain shape. On the other hand, for formations with unilateral responsibility, *persistence*, which is a kind of directed version of rigidity, is also required [1, 14, 15, 16]. When using a directed graph to maintain a formation, it is not enough that the formation be rigid. There can be situations where some interagent distances are correct but it is impossible to satisfy the remaining distance constraints. If such a situation cannot happen, the graph is said to be *constraint consistent.* A directed formation is constraint consistent if each agent has 2 or fewer incoming edges. Intuitively, the directed graph of the formation is persistent if, provided that all the agents are trying to satisfy their constraints, they can satisfy these constraints and hence the global structure of the formation is preserved in the sense that when it moves, it moves as a cohesive whole. See the elegant paper by Anderson et al. [1] and the references therein for a detailed explanation of persistence.

1.3. Cycles

Cycles in formation networks are beneficial for the fact that they provide information feedback. Cycles help to avoid the disturbance rejection problems associated with leader–follower architectures and are more robust to loss of individual links. Agents moving in front of a formation will be aware of the movements of the agents at the back provided that network cycles include agents at different locations in a formation. Moreover, cycles help to control the accumulation of small errors into large deformation.

Fidan and Anderson, in [17], consider a decentralized control problem related to cyclic formation maintenance and interagent distance-keeping for autonomous robot and vehicle platoons in 1-dimensional space. Anderson et al., in [18], consider a formation of 3-point agents moving in the plane, where the agents have a cyclic ordering with each agent required to maintain a nominated distance from its neighbor. The work in [19] considers how to construct decentralized nonlinear control laws to be operated at holonomic point agents that will restore the shape of the formation in the presence of small distortions from the nominal shape even when cycles are present.

The first part of this paper examines the problem domain sketched above and describes the types of sensing and control architectures that are needed to maintain the shape of a formation while the formation moves as a cohesive whole. In other words, the focus is on the network-level question of defining the architecture behind the control laws. Persistent digraph operations on network topology are given focus to find out under what conditions cycles are generated in formations. The $3-$ and $4-$cycle-free rigid and persistent formations were described in [20, 21]. Here, the conditions of k-cycles generated in persistent formations (for any $k \geq 3$, $k \in \mathbb{Z}^+$) are derived. The key tools used are graph-theoretic.

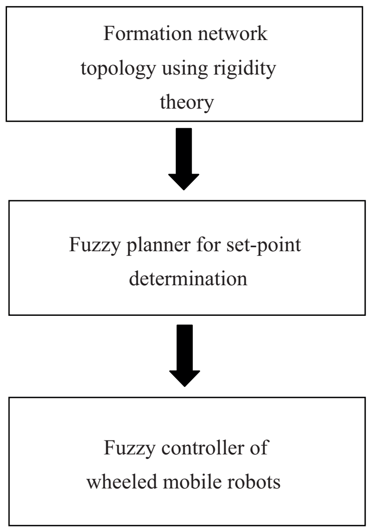

Formation network topology using rigidity theory
Fuzzy planner for set-point determination
Fuzzy controller of wheeled mobile robots

Figure 1. Block diagram of the coordination levels for shape control in multiagent formations: network level, fuzzy planner level, and fuzzy controller level are used in the decomposition of the complex coordination problem into a series of smaller ones.

1.4. Fuzzy planner and fuzzy controller of agents

When a cycle is present in a rigid formation of nonholonomic agents, the control law design problem may not be straightforward. The second part of the paper focuses on the planner level and controller level of the architecture by designing fuzzy logic-based planners and fuzzy logic controllers to be operated at each agent that will maintain the shape of the formation. Here, agents are 2-wheeled kinematic unicycle robots. This permits the decomposition of the complex coordination problem into a series of smaller ones, as shown in Figure 1. Parts of the early work on the fuzzy planner and fuzzy controller were reported in [22]. The scheme for the second half of the paper consists of 2 phases:

Fuzzy planner: On-line set-point determination is achieved by the fuzzy planner. The goal of the agent would be to determine a dynamic reference position for itself in real time that satisfies the prescribed distance constraints and that also integrates well with the fuzzy controller. In this way, the dynamic behavior of the fuzzy controller is modified in order to cope with changes in formation conditions. The idea of applying a fuzzy planner in set-point determination comes from the fact that soft computing techniques are proven to be efficient for multiagent coordination. For example, González et al. [23] proposed the usage of fuzzy modeling to develop individual-oriented models aiming to give some uncertainty to the individual behavior. As a case study, they worked with fish school behavior. The work by Harmati and Skrzypczyk [24] involved a collision-free target tracking problem of a multiagent robot system. A PD-like fuzzy controller tunes the cost function weights directly for the game-theoretic solution and helps to achieve a prescribed value of cost function components. Fuzzy approaches in mobile robot navigation stemming from discrete event systems can be found in [25, 26, 27]. The work by Gu and Yang [28] addressed a multiagent reinforcement learning algorithm with fuzzy policy. Specifically, a leader–follower robotic system and a flocking system were investigated. Using fuzzy rules to design the separation function in flocking algorithms was proposed by Gu and Hu in [29].

Fuzzy controller: The agent moves towards the desired reference position determined by the fuzzy planner starting from its current position. Low-level control of individual kinematic unicycle robots is achieved by fuzzy logic control. A description of how fuzzy logic can be used in autonomous robot navigation can be found in the work done by Saffiotti [30]. Wong et al. [31] proposed a motion control structure with a distance fuzzy controller and an angle fuzzy controller for 2-wheeled mobile robots. They used the PSO algorithm to determine automatically appropriate membership functions of the fuzzy systems. Lee et al. [32] proposed the use of fuzzy logic controllers for 2-wheeled mobile robots that incorporate expert knowledge in terms of linguistic rules. Their simulations and experiments showed that such a fuzzy-controlled wheeled mobile robot can have a better performance than a fine-turned PD-controlled wheeled mobile robot.

Here, a fuzzy design for the controller of the wheeled mobile robot is employed to demonstrate the results on the cyclic network level and the fuzzy planner level. A kinematic model with distance and angle controllers is used.

The rest of this paper is organized as follows: The conditions under which cycles are generated in persistent formations are provided in Section 2. The design of the fuzzy logic planner for set-point determination is explained in Section 3. Fuzzy logic control of wheeled mobile robots is described in Section 4. The simulation results are illustrated on an exemplary cyclic formation in Section 5. The paper ends with some concluding remarks in Section 6.

2. Cycles in the network level of persistent formations

2.1. Terminology

The sensing/communication network topology of a formation with unilateral links is modeled with a digraph $G(V,E)$, where V denotes the set of agents and E denotes the set of sensing/communication links. The order of pairs in E denotes the direction of sensing/communication links. A *directed edge* is written with an ordered pair of end vertices (i,j) representing an edge directed from i to j and drawn with an arrow from i to j, that is, from the leader to the follower. The number of edges directed into i is called the *in-degree* of i and is denoted by $d_G^-(i)$. The number of edges directed out from i is called the *out-degree* of i and is denoted by $d_G^+(i)$. $G=(V,E)$ is called *constraint consistent* if for all $i \in V$, $d_G^-(i) \leq 2$.

2.2. Rigid and persistent formations

A formation is rigid if the fact that all distance constraints are satisfied is sufficient to maintain its shape constant. In Figure 2a, for example, the distances between agents 1 and 2, 2 and 3, 3 and 4, and 4 and 1 are maintained constant, which is not sufficient to maintain the shape if the formation moves in a 2-dimensional space as shown in Figure 2b. On the other hand, constraints in Figure 2c are sufficient to maintain the formation shape. The formation in Figure 2d is rigid, but there is a redundancy. If any one of the edges is removed in the formation in Figure 2d, the shape still can be maintained. For large formations, determining whether a given formation is rigid or not is not is immediate as in the example discussed above.

The combinatorial condition for rigidity in a graph states that a graph $G(V,E)$ is *rigid* if and only if the following hold [33]: (i) There exists a subgraph $G(V,E')$ of $G(V,E)$ where $|E'| = 2|V| - 3$; (ii) for any subgraph $G(V',E'')$ of $G(V,E')$, $E'' \neq \emptyset$, there holds $|E''| \leq 2|V'| - 3$. See the work by Whiteley and Tay and Whiteley [33, 34] for a survey on rigidity and rigid graphs, and the work by Jackson and Jordán [35] for more on its combinatorial aspects.

Rigidity can be applied to formations represented by unilateral links. A digraph is rigid if the undirected graph obtained by forgetting all edge directions is rigid. However, the unilateral character of control actions brings the problem of constraint consistency. Consider, for example, the digraph represented in Figure 3a, which

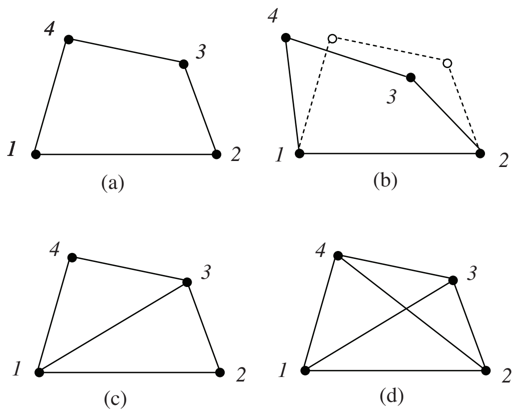


Figure 2. In 2-dimensional space, the formation (a) is not rigid because its representation can be deformed without changing the distance along the edges as represented in (b). Such a deformation could not occur to the formation (c), which is therefore rigid. There is a redundant edge in the rigid formation (d).

represents a formation moving in a 2-dimensional space. It satisfies conditions (i) and (ii) above, and so it is rigid. Observe, however, that agent 3 is responsible for the maintenance of 3 constraints, while agents 1 and 2 are responsible for only 1 constraint. Each of them has to remain at a constant distance from agent L. Agent L itself has no constraint and is called the global leader. Suppose that, starting from a reference position in which all constraints are satisfied, 2 moves in the plane, while keeping its distance to L constant as represented in Figure 3b. Agent 3 is then unable to simultaneously satisfy the 3 constraints for which it is responsible. On the other hand, agents 1 and 2 satisfy all their constraints. Although the graph is rigid, the formation shape is not always maintained, since some constraints are not necessarily satisfied. This is a consequence of the network of unilateral distance constraints, independently of the controllers used. Such a situation never happens with the formation represented in Figure 3c. In summary, the graph in Figure 3a is rigid but not persistent because it is not constraint consistent, while the graph in Figure 3c is both rigid and constraint consistent, making it persistent.

The combinatorial condition for persistence in a digraph states that a digraph $G(V, E)$ is *persistent* if and only if conditions (i) and (ii) for rigidity above hold along with a third condition: (iii) $G(V, E')$ is constraint consistent. The first 2 of these conditions test whether the formation is *rigid* or not. The third condition tests the direction constraints. See, for example, [1, 7, 19, 8] for a detailed explanation of persistence.

2.3. Henneberg construction

The Henneberg construction is a technique for growing persistent graphs, starting from a graph with 2 vertices and 1 edge joining the vertices. At each step in the procedure, either one of the following operations is applied [1, 14]:

Directed 2-valent vertex addition: A vertex q of in-degree 2 is inserted into G by adding the edges (a, q) and (b, q), as shown in Figure 4.

Directed 2-valent vertex deletion: A vertex q of in-degree 2 with the edges (a, q) and (b, q) is deleted from G, as shown in Figure 4 .

Directed 3-valent vertex addition: A vertex q of in-degree 2 and out-degree 1 is inserted into G by deleting the edge (a, b) and adding the edges (a, q), (c, q), and (q, b), as shown in 5. This operation is also

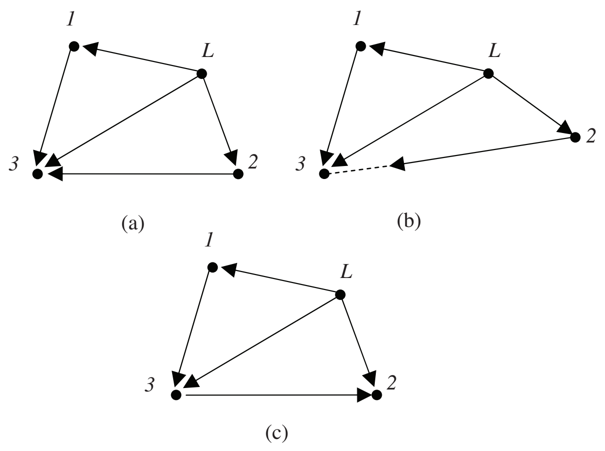


Figure 3. The network in formation (a) is rigid, but the shape of the formation is not necessarily maintained, because all constraints are not necessarily always satisfied. For example, agent 2 has a single distance constraint to satisfy and can freely move on a circle centered on agent L. If it does so, it becomes impossible for 3 to simultaneously satisfy the 3 constraints for which it is responsible, namely $(L,3),(1,3),and(2,3)$, while the other agents satisfy all their constraints, as represented in (b). This situation never happens with the formation represented in (c). In short, graph (a) is rigid but not persistent because it is not constraint consistent, while (c) is both rigid and constraint consistent, making it persistent.

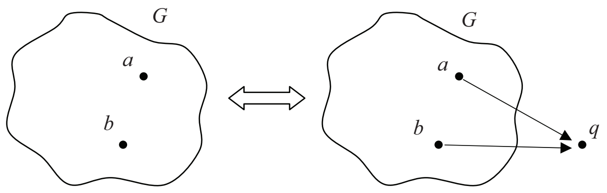


Figure 4. Application of directed 2-valent vertex addition operation is shown from left to right. Application of directed 2-valent vertex deletion operation is shown from right to left.

known as the *edge-splitting* operation.

Directed 3-valent vertex deletion: A vertex q of in-degree 2 and out-degree 1 is deleted from G by deleting the edges (a,q), (c,q), and (q,b) and inserting 1 of the pairs in $\{(a,b),(c,b)\}$, as shown in Figure 5.

Edge reversal: An edge (a,b) in G is replaced by (b,a).

2.4. Cycles in formations

A *directed path* is a nonempty digraph $P=(V,E)$ of the form $V=\{x_0,x_1,\ldots,x_k\}$, $E=\{(x_0,x_1)$, (x_1,x_2), $\ldots$, $(x_{k-1},x_k)\}$. A path is referred to by the sequence of its vertices as $P=x_0x_1\ldots x_k$ and calling P a path from x_0 to x_k. In a *simple path* all the x_i are distinct. In a *closed path*, $x_0=x_k$. The *length of a path* is the number of edges that the path uses. The *distance* between vertices v and w, denoted by $d(v,w)$, is the length of the shortest path directed from v to w. A closed directed path, with no repeated vertices other than the starting and ending vertices, is called a *(directed) cycle* and is denoted by C. The *length of a cycle* C, denoted by $\ell(C)$, is its number of edges. The cycle of length k is called a k-cycle and denoted by C^k. A *cycle graph* is a graph on n vertices containing a single cycle through all vertices. A *cyclic graph* is a graph containing at

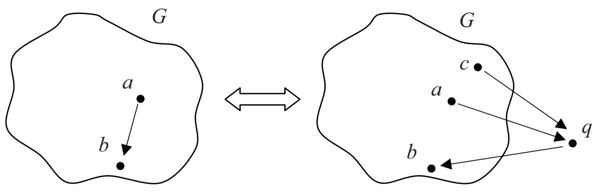


Figure 5. Application of directed 3-valent vertex addition operation is shown from left to right. Application of directed 3-valent vertex deletion operation is shown from right to left.

least one cycle. A graph that is not cyclic is said to be *acyclic*. Now the conditions on k-cycles in persistent digraphs are derived. There is the following theorem:

Theorem 1 *(k-Cycle-free 2-valent vertex addition and deletion) Let a digraph $G = (V, E)$ be given. A vertex q of in-degree 2 is inserted into G by adding the edges (a, q) and (b, q). Let $G^* = (V^*, E^*)$ denote the resultant digraph, i.e. $V^* = V \cup \{q\}$ and $E^* = E \cup \{(a, q), (b, q)\}$. $G = (V, E)$ is k-cycle-free persistent if and only if $G^* = (V^*, E^*)$ is k-cycle-free persistent.*

Proof First, assume that $G = (V, E)$ is k-cycle-free persistent. All newly added paths include q and end in q, since both (a, q) and (b, q) are directed into q. Any new path ending in q does not initiate from q, since the out-degree of q is zero. Therefore, the resultant digraph $G^* = (V^*, E^*)$ does not include a k-cycle. Furthermore, G^* is persistent since the operation is 2-valent vertex addition. Thus, $G^* = (V^*, E^*)$ is k-cycle-free and persistent.

Now assume that $G^* = (V^*, E^*)$ is k-cycle-free persistent. When the vertex q and its incident edges (a, q) and (b, q) are removed, the resultant digraph is still persistent from 2-valent vertex deletion operation. Furthermore, deleting edges does not create a new cycle of any length. Thus, the resultant digraph $G = (V, E)$ is k-cycle-free persistent. $\square$

Now consider 3-valent vertex addition and deletion operations for k-cycle-free persistent digraphs. First, removing a vertex of degree 3 is considered. There is the following theorem:

Theorem 2 *(k-Cycle-free removal of 3-valent agent) Let $G^* = (V^*, E^*)$ be a persistent digraph, where $V^* = V \cup \{q\}$ and $E^* = E \cup \{(a, q), (c, q), (q, b)\}$. Assume that vertex q and its incident edges are removed. One of the edges in $\{(a, b), (c, b)\}$ is inserted. The resulting graph $\hat{G}(V, \hat{E})$, where $\hat{E} = E \cup$ {one of the edges in $\{(a, b), (c, b)\}$}, is a k-cycle-free minimally persistent digraph if one of the following conditions hold: (i) if (a, b) is chosen and the resulting graph is persistent, then in $\hat{G}$, $d(b, a) \neq k-1$; (ii) if (c, b) is chosen and the resulting graph is persistent, then in $\hat{G}$, $d(b, c) \neq k-1$.*

Proof By 3-valent vertex deletion operation, if either of the edges in $\{(a, b), (c, b)\}$ is inserted, at least one of the resulting graphs is persistent. If in $\hat{G}$, $d(b, a) \neq k - 1$, then inserting (a, b) does not create a cycle of length k, because if a cycle C occurs, then $\ell(C) = d(b, a) + 1 \neq k$. Similarly, if $d(b, c) \neq k - 1$, then inserting (c, b) does not create a cycle of length k, because if a cycle C occurs, then $\ell(C) = d(b, c) + 1 \neq k$. $\square$

Now 3-valent vertex addition in a persistent digraph is examined.

Theorem 3 *(k-Cycle-free 3-valent agent addition) Let $G = (V, E)$ be a persistent digraph. Assume that $(a, b) \in E$. Now delete (a, b) and obtain the graph $\hat{G} = (V, \hat{E})$. A vertex q of degree 3 is added to $\hat{G}$ by inserting (a, q), (q, b), and a third edge (c, q) between c and q where $c \in V$. Let the resultant digraph be denoted by $G^*(V^*, E^*)$ where $V^* = V \cup \{q\}$ and $E^* = \hat{E} \cup \{(a, q), (q, b), (c, q)\}$. Then there is no new k-cycle in persistent digraph G^* if the following conditions hold: (i) in $\hat{G}$, $d(b, a) \neq k - 2$; (ii) in $\hat{G}$, $d(b, c) \neq k - 2$.*

Proof Assume that by inserting (a, q) and (q, b), a new cycle $C = (q, b, \ldots, a, q)$ is generated. (If no cycle is generated, $d(b, a) = \infty \neq k$, and thus no new $k-$cycle.) The length of the cycle C in G^* would be $1 + d(b, a) + 1$. If in $\hat{G}$, $d(a, b) \neq k - 2$, then in G^*, $\ell(C) = 1 + d(b, a) + 1 \neq k$. Thus, the operation does not result in a k-cycle.

Similarly, assume that by inserting (c, q) and (q, b), a new cycle $C = (q, b, \ldots, c, q)$ is generated. (If no cycle is generated, $d(b, c) = \infty \neq k$, and then no new $k-$cycle.) The length of the cycle C in G^* would be $1 + d(a, c) + 1$. If in $\hat{G}$, $d(a, c) \neq k - 2$, then in G^*, $\ell(C) = 1 + d(b, c) + 1 \neq k$. Thus, the operation does not result in a k-cycle. □

Theorem 4 *(k-Cycle-free edge reversal) Let $G = (V, E)$ be a persistent digraph where $(b, a) \in E$. Now edge reversal operation is applied on (b, a). First remove (b, a) and denote the resultant digraph $G^*(V, E^*)$, where $E^* = E \setminus \{(b, a)\}$. Then insert (a, b) and call the resultant digraph by $\hat{G}(V, \hat{E})$ where $\hat{E} = E^* \cup \{(a, b)\}$. There is no new k-cycle in persistent digraph $\hat{G}$ if the following conditions hold: $d(b, a) \neq k - 1$ in $G^*(V, E^*)$.*

Proof The only way that the resultant graph $\hat{G}$ has a new $k-$cycle comes from the insertion of the new edge (a, b) into $G^*(V, E^*)$. If $d(b, a) \neq k - 1$ in $G^*(V, E^*)$, then the new cycle, which can possibly be generated by the insertion of (a, b), will have a length of $\ell(C) = d(b, a) + 1$. Since, in G^*, $d(b, a) \neq k - 1$, then $\ell(C) \neq k$. Thus, the operation does not result in a k-cycle. □

The following conclusions can be drawn from the theorems above: 2-valent vertex addition or deletion operations do not generate k-cycles. If 2-vertex addition or deletion are used, the resulting formation is always cycle-free. If 3-valent vertex addition or deletion, or edge reversal, are used, care should be taken. The directed distance between the end vertices of the split edge or reversed edge determines whether a new cycle of length k is created or not.

There are 2 general types of networks within persistent formations [1]. In the first, which is called *global-leader network*, there is 1 global leader agent of in-degree 0, 1 follower agent of in-degree 1, and other agents of in-degree 2. In the second , which is called *co-leader network*, all agents have in-degree 2, except 3 agents of in-degree 1.

Theorem 5 *Co-leader networks occur only in cyclic formations.*

Proof Suppose that co-leader networks occur in acyclic formations, too. Acyclic formations can be built by using only 2-valent vertex addition and create topological order of vertices. Apply 2-valent vertex deletion to the formation in reverse order until a single edge is left. There is 1 vertex of in-degree 0 (having 2 degrees of freedom left) and 1 vertex of in-degree 1 (having 1 degree of freedom left). Thus, these 2 agents manage all the degrees of freedom that the formation has, which is 3, leaving no degree of freedom to the rest of the agents. Thus, all the remaining vertices have an in-degree of 2, which means that the acyclic formation has

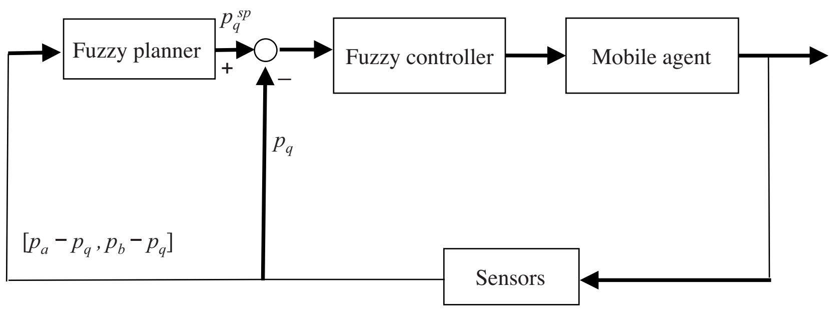


Figure 6. Block diagram showing the fuzzy planner and fuzzy controller. Mobile agent sensors obtain the data $p_a - p_q$ and $p_b - p_q$. The output of the fuzzy controller is the set-point ${p_q}^{sp}$.

global leader architecture. This contradicts with the initial assumption. Thus, co-leader networks occur only in cyclic formations. □

On the other hand, the reverse of Theorem 5 is not true, i.e. cyclic formations can have a global-leader network or a co-leader network. An example of a cyclic co-leader network is a simple 3-cycle. Care should be taken with cyclic co-leader networks. For example, a simple cycle of 3 agents forming a triangle has 3 co-leader agents, each having 1 degree of freedom. These 3 degrees of freedom distributed onto 3 co-leader agents should be managed in a coordinated fashion to make the formation move. As shown above, all co-leader networks have cyclic graphs. If one desires to avoid the co-leader coordination problem in cyclic networks, one can use cyclic global-leader networks, because co-leader networks form only a subset of cyclic formations. Thus, the rest of the cyclic formations have the global-leader network architecture. One such example of cyclic global networks can be seen in the simulation section of the paper.

3. Fuzzy planner for set-point determination

3.1. What is measured by each agent?

It is required that the control be a function only of relative measurements. A typical motion task for one agent moving in a formation with a distributed control law is to maintain a prescribed distance to a certain set of mobile agents. For example, if agent q can see agents a and b, then the measurements available to agent q are $p_a - p_q$ and $p_b - p_q$, where p_q, p_a, and p_b denote the positions of agents q, a, and b, respectively. The set-point is a function of these 2 measurements as shown in Figure 6.

3.2. The task of the fuzzy planner

The fuzzy planner adjusts the set-point of the fuzzy controller according to the sensed data. The controller of an agent needs a unique set-point, which is the desired location for an agent at each time instant as the formation moves through its environment. Each agent determines the set-point of itself in each time interval by its own fuzzy planner. Suppose that an agent q has 2 in-coming links from agents a and b. These 2 links essentially represent 2 distance constraints to be satisfied by agent q. Thus, agent q has to maintain a distance of d_{qa} with respect to agent a and a distance of d_{qb} with respect to agent b. Thus, the fuzzy planner converts the distance measurements of an agent into a single set-point position for its fuzzy controller.

3.3. Components of fuzzy logic planner

The proposed fuzzy planner consists of fuzzy planner subsystems. Each fuzzy logic subsystem has 3 main components. The first block is the fuzzification interface, which converts controller inputs into information that the inference mechanism can easily use to activate and apply rules. The second block is called the rule-base and inference engine. It is a set of *If-Then* rules and contains a fuzzy logic quantification of the linguistic description of how to achieve good control. The inference mechanism emulates the expert's decision-making in interpreting and applying knowledge about how best to control the plant. The third operation is called defuzzification. The resulting fuzzy set is defuzzified into a crisp control signal.

3.4. Input and output membership functions of the fuzzy planner subsystem

In the fuzzy planner here, the input variable of the fuzzy logic unit is distance, d_e. The output variable of the fuzzy planner is the velocity v of the set-point. The values of d_e (Figure 7a) are cropped down to the interval of $[0 \quad 2\times$ *desired distance*$]$. That is, if $d_e > 2$, it is set to 2 and applied as an input into the fuzzy logic unit. d_e is mapped to linguistic variables *DistAgent* by the fuzzification operator. The fuzzy logic unit inputs are composed of the 3 linguistic terms (*Close, Far, Good*). The velocity for the set-point (fuzzy logic unit output) is also partitioned into the 3 fuzzy sets (*forward, back, stop*) in the linguistic variable *Move.* This set of linguistic terms forms a fuzzy partition of input and output spaces. Figures 7a and 7b show the membership functions of the fuzzy sets for the distance and the velocity of the set-point. Note that the center point of the *Good* in *DistAgent* depends on the desired distance between 2 agents. Depending on this desired distance, the centerpiece of this membership function is altered for each distance constraint.

3.5. The type of the fuzzy planner subsystem

Mamdani-type fuzzy inference, which is one of the commonly used rules of fuzzy implication associated with the max–min composition, is used both in the fuzzy planner and the fuzzy controller. The fuzzy output is generated by using the centroid computation method. The trapezoidal and triangular membership functions were used for both input and output of the fuzzy logic unit. A reactive rule-base governing the agent behavior is synthesized from the human heuristics with respect to various situations of the environment. The fuzzy *If-Then* rules for the fuzzy planner are given in Table 1.

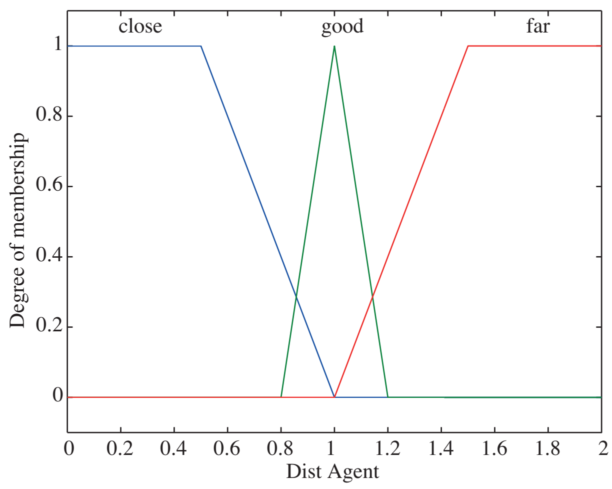
Table 1. Fuzzy logic rules for set-point planner.

If (*DistAgent* is *close*)	Then (*Move* is *back*)
If (*DistAgent* is *far*)	Then (*Move* is *forward*)
If (*DistAgent* is *good*)	Then (*Move* is *stop*)

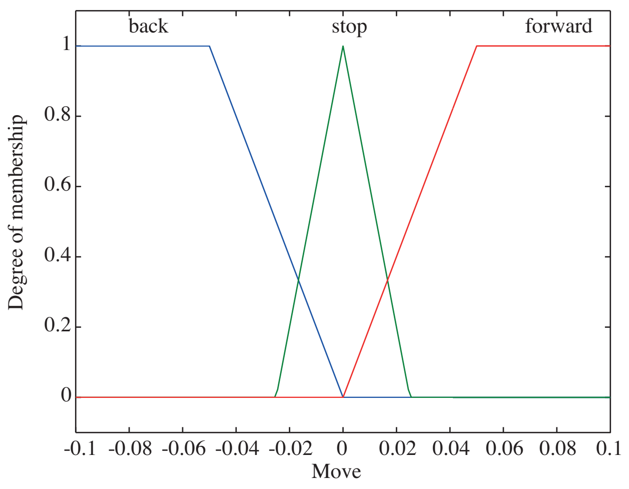
3.6. Combining the outputs of fuzzy logic subsystems

Each fuzzy logic subsystem provides an output velocity for the set-point along the direction of the constraint to be maintained. Next, the details of how a unique set-point is computed from velocity outputs of fuzzy subsystems are explained.

Suppose that the output of a fuzzy planner subsystem is v_{qa}, the vector component along the line connecting agent q and agent a. A separate fuzzy planner subsystem computes v_{qb}, the other vector component along the line connecting agent q and agent b. These 2 vector components are summed and the result is scaled



(a) Input membership functions of the fuzzy planner for set-point determination.



(b) Output member ship functions of the fuzzy planner for set-point determination.

Figure 7. Membership functions of the fuzzy planner for set-point determination.

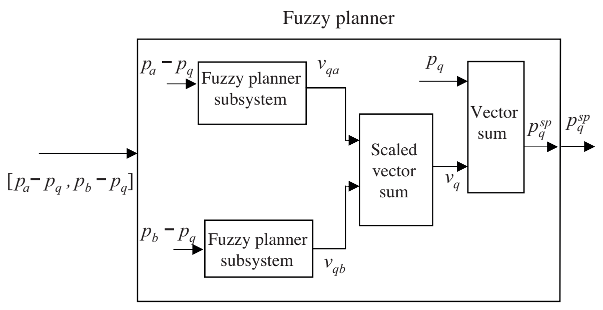


Figure 8. Finding the set points using fuzzy logic. Scaled vector sum limits the maximum velocity of the agent.

not to exceed the maximum possible velocity of an agent as shown in Figure 8. If the velocity violates its limits, it is forced down to its proper values. The resulting vector v_q is added to the current position of q and the result gives the set point position of q, namely ${p_q}^{sp}$. (Observe that, for an agent with only 1 incoming link, the output velocity vector computed by 1 subsystem is the set-point position for that agent. There is one such agent in the cyclic formation examined in the simulation in Section 5.)

4. Fuzzy controller of unicycle 2-wheeled mobile robot

The robots are 2-wheeled vehicles with sensors that allow them to measure the relative positions of some of the other robots. The simplest model for a wheeled vehicle is the "kinematic unicycle" with a small castor ball at the front end to establish its mechanical stability.

The kinematic model of the system is:

$$x = v \cos \phi \tag{1}$$

$$y = v \sin \phi \tag{2}$$

$$\dot{\phi} = \omega \tag{3}$$

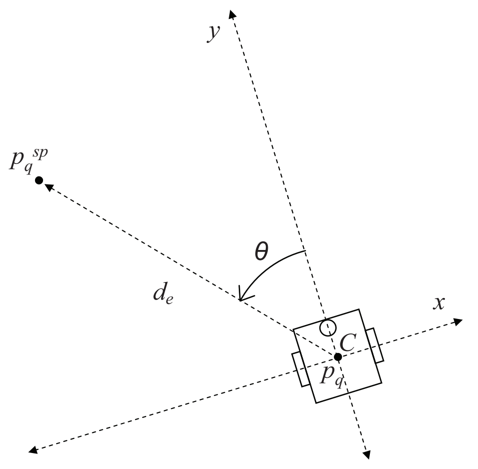


Figure 9. Robot-centric coordinate system.

where v and ω are the translational and the angular velocities of the unicycle, respectively.

In the robot-centric view, the robot accounts for the relative position of the set-point with respect to itself. Therefore, the position of the robot is always the origin when a new set-point is given, as shown in Figure 9.

4.1. Fuzzy controller membership functions and the rule base

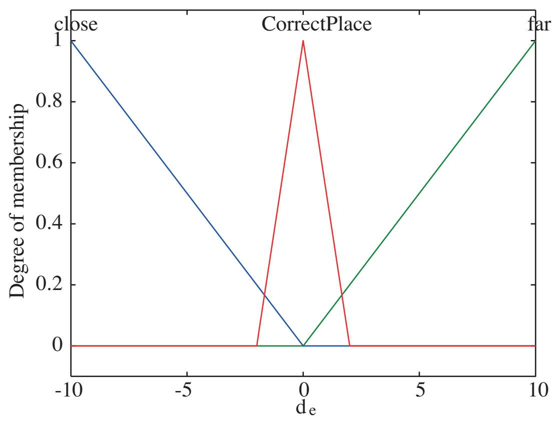
The Mamdani-type fuzzy controller has the input d_e and θ. Here $d_e = |p_q - {p_q}^{sp}|$, where p_q is the position of agent q and ${p_q}^{sp}$ is the set-point of agent q. θ is the counter-clockwise angle between the y-axis of the robot-centric coordinate frame and the line connecting the center of the robot to the set-point position, as shown in Figure 9. The fuzzy controller has the outputs v and $\dot{\theta}$, which are the translational and angular velocities of the unicycle robot. Input and output membership functions of this fuzzy controller are shown in Figures 10a–10d. For the controller, the universe of discourse for distance is chosen to be $[-10, 10]$. If the actual distance is bigger than 10, it is forced down to 10. Fuzzy rules are listed in Table 2. Rules are the most widespread form of knowledge representation in intelligent systems. Their popularity is explained by their simplicity and transparency from both a theoretical and a practical point of view. This implies that rule sets are relatively easy to handle and investigate.

5. Simulation results

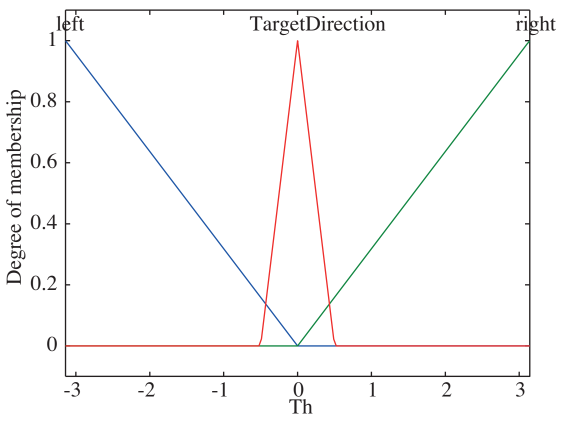
The fuzzy planner and the fuzzy controller, incorporated with an exemplary cyclic network topology generated by the results obtained in the first half of the paper, have been implemented as a simulation experiment on kinematic unicycle mobile robots. All simulation results have been performed using the MATLAB environment.

Table 2. Fuzzy logic rules for the controller.

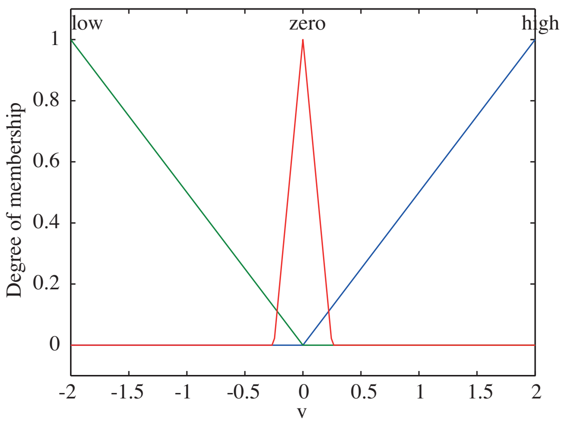
If (d_e is *close*)	Then (v is *low*)
If (θ is at *Right*)	Then ($\dot{\theta}$ is *TurnRight*)
If (d_e is *far*)	Then (v is *high*)
If (θ is at *left*)	Then ($\dot{\theta}$ is *TurnLeft*)
If (d_e is *CorrectPlace*)	Then (v is *zero*)
If (θ is *TargetDirection*)	Then ($\dot{\theta}$ is *zero*)



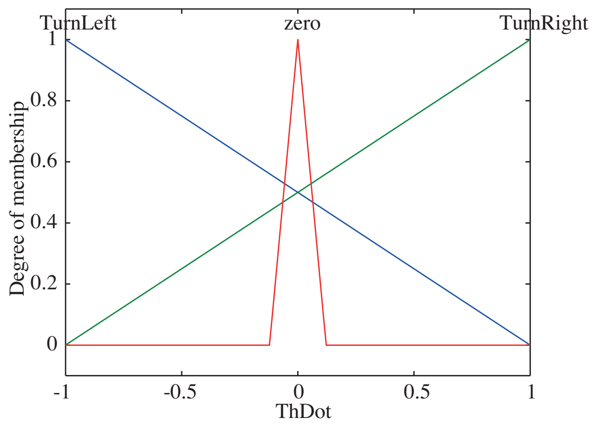
(a) Input membership functions of the fuzzy controller for the variable d_e, which is the distance between the current position of the unicycle and the set-point position.



(b) Input member ship functions of the fuzzy controller for the variable θ, which is the counter-clockwise angle between the y-axis of the robot-centric coordinate frame and the line connecting the center of the robot with the set-point position.



(c) Output membership functions of the fuzzy controller for the variable v, the translational velocity of the unicycle robot.



(d) Output member ship functions of the fuzzy controller for the variable $\dot{\theta}$, the angular velocity of the unicycle robot.

Figure 10. Membership functions of the fuzzy controller.

The correct functioning of the model can be analyzed at 2 levels. One aspect is to verify that the simulated formation will reproduce constant distances between agents among which interagent distances are explicitly maintained by means of sensing/communication links. The second aspect is to verify that the remaining interagent distances (the distances between agents among which there are no sensing/communication links) remain constant.

A cyclic persistent formation of 6 agents, which includes 1 cycle, is simulated during 5000 iterations to test shape control on the network topology. An instantiation of this formation is shown in Figure 11. The underlying rigid graph of this formation is known to have no acyclic persistent orientation [16]. That is, no acyclic choice of directed constraints will satisfy the persistence conditions on this formation. Thus, cyclic orientation on this formation is inevitable. The robot with in-degree 0 (global leader) is labeled with L and the robot with in-degree 1 (the first follower) is labeled with 1. The rest of the robots have an in-degree of 2. The global leader is moving on a prescribed trajectory.

The formation is generated by a sequence of 2-valent and 3-valent vertex additions as follows: start with

2 vertices L and 1, and 1 edge $(L,1)$; insert vertex 2 by 2-valent vertex addition using the edges $(L,2)$ and $(1,2)$; insert vertex 3 by 2-valent vertex addition using edges $(L,3)$ and $(1,3)$; insert vertex 4 by 3-valent vertex addition by removing edge $(1,3)$ and inserting the edges $(1,4)$, $(2,4)$, and $(4,3)$; finally, insert vertex 5 using 3-valent vertex addition by removing the edge $(1,4)$ and inserting the edges $(1,5)$, $(3,5)$, and $(5,4)$ and obtain the resultant formation network. It has a cycle of length 3, $(4,3,5,4)$. The cycle is generated in the final step of the Henneberg construction.

The trajectories of robots are shown in Figure 12. As the global leader moves in 2-dimensional space, the remaining agents act to satisfy their distance constraints, which results in the preservation of the formation shape as a consequence of persistence. This can be seen in Figures 13a and 13b. By explicitly maintaining some interagent distances constant as in Figure 13a, all the remaining interagent distances remain constant as in Figure 13b, as predicted by the persistence of the network. Small variations in constant interagent distances are influenced by the nonholonomic constraints of the kinematic unicycle.

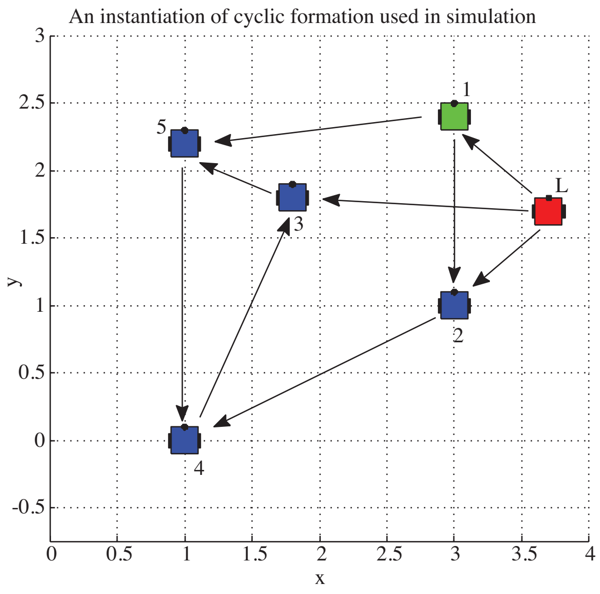


Figure 11. An exemplary cyclic persistent formation of 6 agents, which includes a cycle. The robot with in-degree 0 (global leader) is labeled with L and in red. The robot with in-degree 1 (the first follower) is labeled with 1 and in green. The rest of the robots have an in-degree of 2 and are in blue.

6. Concluding remarks

Cyclic persistent networks are worth studying, not only for theoretical completeness but also because the presence of cycles provide the possibility of information feedback. Information feedback would be desirable in formations for 2 reasons: (i) to help leaders be aware of the loss or failure of their following agents; and (ii) to control the accumulation of small errors into large deformation.

In this article, the shape control problem of a group of agents in cyclic networks using a decentralized fuzzy logic-based planner and fuzzy logic-based control was considered. In the first part of the paper, the conditions under which cycles are generated in persistence operations were provided. The second part of the paper introduced the fuzzy planner and made use of the fuzzy controller, showing that they are suitable for real-

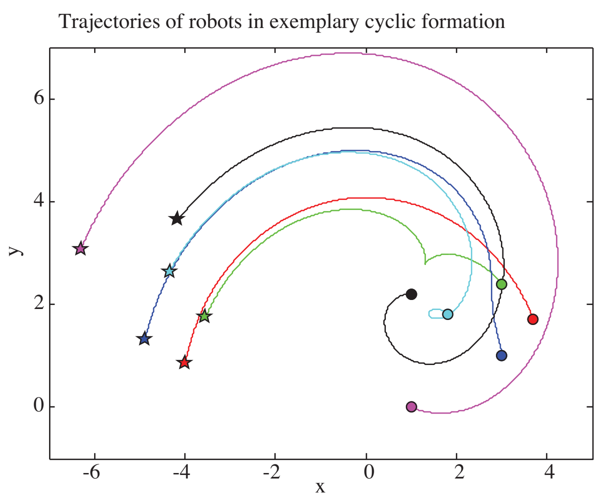
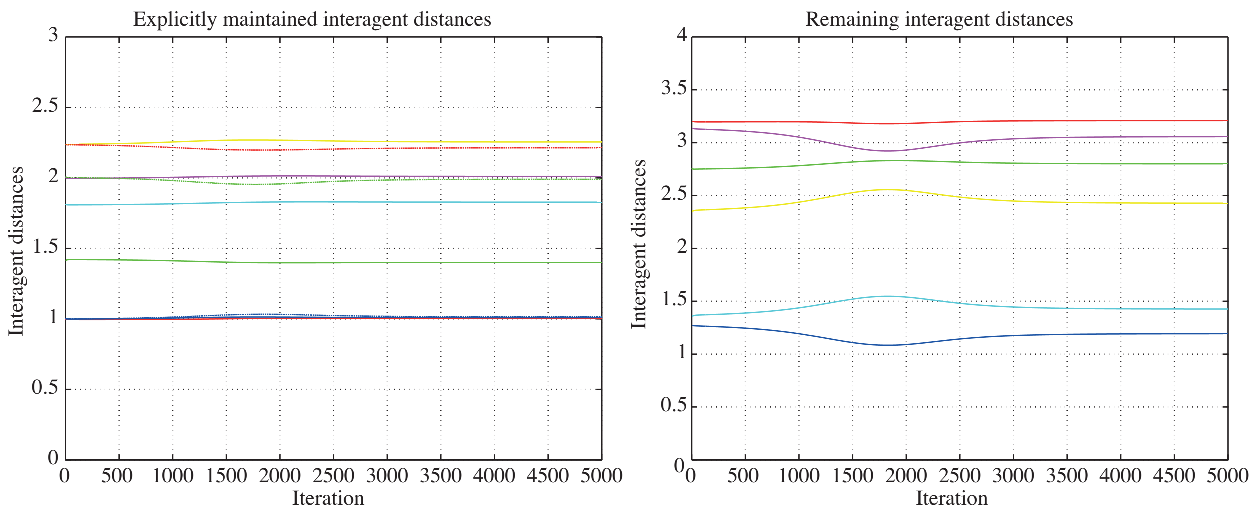


Figure 12. Trajectories of robots as the formation evolves in 2-dimensional space. The initial positions of robots are shown with circles; the final positions are shown with stars. The global leader (robot with in-degree 0) is depicted in red while the first follower (robot with in-degree 1) is depicted in green. All the other robots have in-degree 2.



(a) The distances between agents among which interagent distances are explicitly maintained by means of sensing/communication links, as the formation evolves in two-dimensional space. These agent pairs are: $\{(L, 1), (L, 2), (1, 2), (L, 3), (1, 5), (2, 4), (3, 5), (5, 4), (4, 3)\}$

(b) The remaining inter-agent distances (the distances between agents among which there are no sensing/communication links), as the formation evolves in two-dimensional space. The remaining agent pairs are: $\{(L, 4), (L, 5), (1, 3), (1, 4), (2, 3), (2, 5)\}$.

Figure 13. Interagent distances as the formation evolves in 2-dimensional space.

time control in multiagent applications. The validity of the proposed approach was verified through simulation experiments.

Acknowledgments

This work was supported by the Scientific and Technological Research Council of Turkey through grant 108E189 and was carried out in the framework of European Commission COST Action IC0806, "Intelligent Monitoring, Control and Security of Critical Infrastructure Systems."

References

[1] B.D.O. Anderson, C. Yu, B. Fidan, J.M. Hendrickx, "Rigid graph control architectures for autonomous formations", Control Systems Magazine, IEEE, Vol. 28, pp. 48–63, 2008.

[2] Y.H. Esin, M. Ünel, "Formation control of nonholonomic mobile robots using implicit polynomials and elliptic Fourier descriptors", Turkish Journal of Electrical Engineering & Computer Sciences, Vol. 18, pp. 765–780, 2010.

[3] S.B. Akat, V. Gazi, L. Marques, "Asynchronous particle swarm optimization-based search with a multi-robot system: simulation and implementation on a real robotic system", Turkish Journal of Electrical Engineering & Computer Sciences, Vol. 18, pp. 749–764, 2010.

[4] T. Eren, P.N. Belhumeur, B.D.O. Anderson, A.S. Morse, "A framework for maintaining formations based on rigidity", in Proceedings of the 15th IFAC World Congress, Barcelona, Spain, pp. 2752–2757, 2002.

[5] T. Eren, B.D.O. Anderson, A.S. Morse, W. Whiteley, P.N. Belhumeur, "Operations on rigid formations of autonomous agents", Communications in Information and Systems, Vol. 3, pp. 223–258, 2004.

[6] C. Yu, J.M. Hendrickx, B. Fidan, B.D.O. Anderson, V.D. Blondel, "Three and higher dimensional autonomous formations: rigidity, persistence and structural persistence", Automatica, Vol. 43, pp. 387–402, 2007.

[7] B. Fidan, C. Yu, B.D.O. Anderson, "Acquiring and maintaining persistence of autonomous multi-vehicle formations", IET Control Theory and Applications, Vol. 1, pp. 452–460, 2007.

[8] J.M. Hendrickx, B.D.O. Anderson, J.C. Delvenne, V.D. Blondel, "Directed graphs for the analysis of rigidity and persistence in autonomous agents systems", International Journal of Robust and Nonlinear Control, Vol. 17, pp. 960–981, 2007.

[9] L. Krick, M.E. Broucke, B.A. Francis, "Stabilisation of infinitesimally rigid formations of multi-robot networks", International Journal of Control, Vol. 82, pp. 423–439, 2009.

[10] T. Eren, D. Goldenberg, W. Whiteley, A. Morse, B.D.O. Anderson, P. Belhumeur, "Rigidity, computation, and randomization in network localization", in Proceedings of the International Annual Joint Conference of the IEEE Computer and Communications Societies, Hong Kong, Vol. 4, pp. 2673–2684, 2004.

[11] J. Aspnes, T. Eren, D. Goldenberg, W. Whiteley, Y.R. Yang, A.S. Morse, B.D.O. Anderson, P.N. Belhumeur, "A theory of network localization", IEEE Transactions on Mobile Computing, Vol. 5, pp. 1663–1678, 2006.

[12] B.D.O. Anderson, P.N. Belhumeur, T. Eren, D.K. Goldenberg, A.S. Morse, W. Whiteley, Y.R. Yang, "Graphical properties of easily localizable sensor networks", Wireless Networks, Vol. 15, pp. 177–191, 2009.

[13] Y. Dieudonne, O. Labbani-Igbida, F. Petit, "Deterministic robot-network localization is hard", IEEE Transactions on Robotics, Vol. 26, pp. 331–339, 2010.

[14] T. Eren, W. Whiteley, B.D.O. Anderson, A.S. Morse, P.N. Belhumeur, "Information structures to secure control of rigid formations with leader-follower architecture", in Proceedings of the American Control Conference, Portland, Oregon, Vol. 4, pp. 2966–2971, 2005.

[15] B.D.O. Anderson, C. Yu, B. Fidan, J.M. Hendrickx, "Control and information architectures for formations", in Proceedings of the IEEE Conference on Control Applications, pp. 1127–1138, 2006.

[16] J. Bang-Jensen, T. Jordán, "On persistent directed graphs", Networks, Vol. 52, pp. 271–276, 2008.

[17] B. Fidan, B.D.O. Anderson, "Switching control for robust autonomous robot and vehicle platoon formation maintenance", in Proceedings of the 15th Mediterranean Conference on Control and Automation, Athens, Greece, pp. 1–6, 2007.

[18] B.D.O. Anderson, C. Yu, S. Dasgupta, A.S. Morse, “Control of a three-coleader formation in the plane”, Systems and Control Letters, Vol. 56, pp. 573–578, 2007.

[19] C. Yu, B.D.O. Anderson, S. Dasgupta, B. Fidan, “Control of minimally persistent formations in the plane”, SIAM Journal on Control and Optimization, Vol. 48, pp. 206–233, 2009.

[20] T. Eren, “3-Cycle-free rigidity for multi-agent control systems”, in Proceedings of the 17th Mediterranean Conference on Control and Automation, Thessaloniki, Greece, pp. 258–263, 2009.

[21] T. Eren, “3-Cycle-free persistence for cooperative control of formations with acyclic control structure”, in Proceedings of the ICONS 2009, 2nd IFAC International Conference on Intelligent Control Systems and Signal Processing, İstanbul, Turkey, pp. 1–6, 2009.

[22] T. Eren, “Using fuzzy logic in shape control of cyclic networks in multi-vehicle formations”, in Proceedings of the 19th Mediterranean Conference on Control and Automation, Corfu, Greece, pp. 1466–1471, 2011.

[23] J.C. González, C. Dalforno, R. Suppi, E. Luque, “A fuzzy logic fish school model”, Springer Lecture Notes in Computer Science, Vol. 5544, pp. 13–22, 2009.

[24] I. Harmati, K. Skrzypczyk, “Robot team coordination for target tracking using fuzzy logic controller in game theoretic framework”, Robotics and Autonomous Systems, Vol. 57, pp. 75–86, 2008.

[25] Y. Boutalis, K. Schmidt, “Multi-objective decision making using fuzzy discrete event systems: a mobile robot example”, in Proceedings of the Mediterranean Conference on Control and Automation, Marrakesh, Morocco, pp. 575–580, 2010.

[26] G. Tampakis, K. Schmidt, Y. Boutalis, “Mobile robot navigation based on fuzzy discrete event systems”, in Proceedings of the Conference on Intelligent Control Systems and Signal Processing, İstanbul, Turkey, pp. 1–6, 2009.

[27] R. Huq, G.K.I. Mann, R.G. Gosine, “Behavior modulation technique in mobile robotics using fuzzy discrete event system”, IEEE Transactions on Robotics, Vol. 22, pp. 903–916, 2006.

[28] D. Gu, E. Yang, “Fuzzy policy reinforcement learning in cooperative multi-robot systems”, Journal of Intelligent & Robotic Systems, Vol. 48, pp. 7–22, 2007.

[29] D. Gu, H. Hu, “Fuzzy logic to design separation function in flocking algorithms”, IEEE Transactions on Fuzzy Systems, Vol. 16, pp. 826–838, 2008.

[30] A. Saffiotti, “The uses of fuzzy logic in autonomous robot navigation”, Soft Computing, Vol. 1, pp. 180–197, 1997.

[31] C. Wong, H. Wang, S. Li, “PSO-based motion fuzzy controller design for mobile robots”, International Journal of Fuzzy Systems, Vol. 10, pp. 24–32, 2008.

[32] T. Lee, F.H.F. Leung, P.K.S. Tam, “Position control for wheeled mobile robots using a fuzzy logic controller”, in Proceedings of the IEEE International Conference on Industrial Electronics Society, Vol. 2, pp. 525–528, 1999.

[33] W. Whiteley, “Rigidity and scene analysis”, in Handbook of Discrete and Computational Geometry, J. Goodman , J. O’Rourke, Eds., CRC Press, Boca Raton, FL, USA, pp. 893–916, 1997.

[34] T. Tay, W. Whiteley, “Generating isostatic frameworks”, Structural Topology, Vol. 11, pp. 21–69, 1985.

[35] B. Jackson, T. Jordán, “Connected rigidity matroids and unique realizations of graphs”, Journal of Combinatorial Theory, Series B, Vol. 94, pp. 1–29, 2005.