Available online at www.sciencedirect.com
ScienceDirect
Robotics and Autonomous Systems 56 (2008) 324–333
Robotics and Autonomous Systems
www.elsevier.com/locate/robot

Distributed coordination architecture for multi-robot formation control

Wei Ren*, Nathan Sorensen

Department of Electrical and Computer Engineering, Utah State University, Logan, UT 84322-4120, USA

Received 23 October 2006; received in revised form 17 August 2007; accepted 17 August 2007
Available online 31 August 2007

Abstract

In the exploration and implementation of formation control strategies, communication range and bandwidth limitations form a barrier to large scale formation control applications. The limitations of current formation control strategies involving a leader–follower approach and a consensus-based approach with fully available group trajectory information are explored. A unified, distributed formation control architecture that accommodates an arbitrary number of group leaders and arbitrary information flow among vehicles is proposed. The architecture requires only local neighbor-to-neighbor information exchange. In particular, an extended consensus algorithm is applied on the group level to estimate the time-varying group trajectory information in a distributed manner. Based on the estimated group trajectory information, a consensus-based distributed formation control strategy is then applied for vehicle level control. The proposed architecture is experimentally implemented and validated on a multi-robot platform under local neighbor-to-neighbor information exchange with a single or multiple leaders involved.

Keywords: Formation control; Consensus algorithm; Coordination; Multi-robot systems

1. Introduction

In the field of cooperative control, approaches for achieving of formation maintenance among multiple vehicles have received significant attention. Given the limitations of communication bandwidth and communication range in many applications, the need for distributed algorithms that require only local neighbor-to-neighbor information exchange is apparent.

A typical leader–follower formation control approach (e.g., [1]) assumes only one group leader within the team. In this case, only the group leader has the knowledge of group trajectory information, which is either preprogrammed in the group leader or provided to the group leader by an external source. The formation is then built on the reaction of the other group members to the motion of the group leader. The fact that only a single group leader is involved in the team implies that the leader–follower approach is simple to implement and understand, and the requirement on communication bandwidth is reduced. This is, however, a single point of massive failure type system because the loss of the group leader causes the entire group to fail. Another issue with the typical leader–follower approach is the lack of inter-vehicle information feedback throughout the group. For example, feedback from the followers is not used by the leader so the formation can become disjoint and followers can be left behind if they are not able to track the motion of the leader accurately.

In order to overcome this type of single point of failure tendency, much research has been focusing on decentralized or distributed cooperative control strategies where vehicle control laws are coupled and each vehicle makes its own decision according to the states of its neighbors (e.g., [2–17]). This allows the group to continue on to achieve an objective even in the presence of failure of any group member.

Among the decentralized or distributed cooperative control strategies, consensus algorithms (e.g., [6–12,15]) focus on driving the information states of all vehicles to a common value. For formation stabilization with a static formation centroid, if each vehicle in a group can reach consensus on the center point of the desired formation and specify a corresponding desired deviation from the center point, then vehicle formations can be achieved. To apply consensus algorithms to achieve formation maneuvering with a time-varying formation centroid trajectory, either the common formation velocity for the group or the desired group trajectory is assumed to be known by

* Corresponding author. Tel.: +1 435 797 2831; fax: +1 435 797 3054.
E-mail address: wren@engineering.usu.edu (W. Ren).

0921-8890/$ - see front matter
doi:10.1016/j.robot.2007.08.005

each vehicle in the group as in [3,5,17,18]. In particular, [5] assumes that a sequence of constant, desired formation centroid states are preprogrammed on each vehicle. However, this approach cannot account for dynamically changing formation centroid states in response to dynamically changing situational awareness. While a flocking behavior is achieved in [4,19] when no vehicle has the knowledge of group formation velocity, an accurate formation geometry is not specified. In this paper, we focus on applications that require accurate formation geometry maintenance with desired group trajectory information involved.

The requirement that each vehicle have the knowledge of the desired group trajectory may not be realistic for many applications. For example, communication bandwidth and range limitations may prevent each vehicle in the group having access to the group trajectory information. Also, to increase stealth and flexibility, only a portion of the vehicles in the team may be provided with the desired group trajectory information. In addition, it is also possible that only a portion of the vehicles are able to detect a target or dangerous source at a certain time instant, and those vehicles in turn serve as the group leaders to guide the behaviors of the other group members.

Given the strength of the consensus algorithms for formation control with coupling involved between neighboring vehicles and the effectiveness of a traditional leader–follower approach when group trajectory information is limited in the formation, integrating the two approaches yields the strength of both strategies. Bandwidth limitations for the group can be handled in limiting the amount of group trajectory information availability within the group, while robustness is achieved with distributed nature of the consensus algorithms.

The main contributions of the current paper are twofold. First, we propose a unified, distributed formation control architecture that accommodates an arbitrary number of group leaders and allows for arbitrary information flow among vehicles without adding complexity to the control law design and analysis. In particular, an extended consensus algorithm is applied on the group level to estimate the time-varying group trajectory information in a distributed manner. Based on the estimated group trajectory information, a consensus-based distributed formation control strategy is then applied for vehicle level control. Second, the proposed formation control architecture is experimentally implemented and validated on a multi-robot platform and the results are discussed. It is worthwhile to mention that although various strategies for decentralized or distributed formation control have been studied in the literature, few have been systematically verified on experimental platforms. A preliminary version of the work has been presented at the 2007 American Control Conference [20].

2. Background and preliminaries

2.1. Graph theory notations

It is natural to model information exchange among vehicles by directed or undirected graphs. A *digraph* (*directed graph*) consists of a pair $(\mathcal{N}, \mathcal{E})$, where $\mathcal{N}$ is a finite nonempty set of nodes, and $\mathcal{E} \in \mathcal{N} \times \mathcal{N}$ is a set of ordered pairs of nodes, called edges. An *edge* (i, j) in a digraph denotes that vehicle j can obtain information from vehicle i, but not necessarily vice versa. In contrast, the pairs of nodes in an *undirected graph* are unordered, where an edge (i, j) denotes that vehicles i and j can obtain information from one another. Note that an undirected graph can be considered a special case of a digraph, where an edge (i, j) in the undirected graph corresponds to edges (i, j) and (j, i) in the digraph. If there is an edge from node i to node j in a digraph, then i is the *parent node*, and j is the *child node*. A *directed path* is a sequence of edges of the form $(v_{i_1}, v_{i_2}), (v_{i_2}, v_{i_3}), \ldots$, where $v_{i_j} \in \mathcal{N}$, in a digraph. An *undirected path* in an undirected graph is defined analogously. In a digraph, a *cycle* is a directed path that starts and ends at the same node. A digraph is *strongly connected* if there is a directed path from every node to every other node. An undirected graph is *connected* if there is a path between any distinct pair of nodes. A *directed tree* is a digraph, where every node has exactly one parent except for one node, called the *root*, which has no parent, and the root has a directed path to every other node. Note that in a directed tree, each edge has a natural orientation away from the root, and no cycle exists. In the case of undirected graphs, a *tree* is a graph in which every pair of nodes is connected by exactly one path. A *directed spanning tree* of a digraph is a directed tree formed by graph edges that connect all of the nodes of the graph. A graph *has* or *contains a directed spanning tree* if there exists a directed spanning tree being a subset of the graph. Note that the condition that a digraph has a directed spanning tree is equivalent to the case that there exists at least one node having a directed path to all of the other nodes. In the case of undirected graphs, having an *undirected spanning tree* is equivalent to being connected. However, in the case of directed graphs, having a directed spanning tree is a weaker condition than being strongly connected.

The *adjacency matrix* $A = [a_{ij}] \in \mathbb{R}^{n\times n}$ of a digraph is defined as $a_{ii} = 0$ and $a_{ij} > 0$ if $(j, i) \in \mathcal{E}$ where $i \neq j$. The adjacency matrix of an undirected graph is defined analogously except that $a_{ij} = a_{ji}$, $\forall i \neq j$, since $(j, i) \in \mathcal{E}$ implies $(i, j) \in \mathcal{E}$. Let matrix $L = [\ell_{ij}] \in \mathbb{R}^{n\times n}$ be defined as $\ell_{ii} = \sum_{j\neq i} a_{ij}$ and $\ell_{ij} = -a_{ij}$, where $i \neq j$. The matrix L satisfies the following conditions:

$$\ell_{ij} \leq 0, i \neq j, \quad \sum_{j=1}^{n} \ell_{ij} = 0, \quad i = 1, \ldots, n. \tag{1}$$

For an undirected graph, L is called the *Laplacian matrix* [21], which is symmetric positive semi-definite. However, L for a digraph does not have this property.

Let $\mathbf{1}$ and $\mathbf{0}$ denote the $n \times 1$ column vector of all ones and all zeros respectively. Let I_n denote the $n \times n$ identity matrix. Let $M_n(\mathbb{R})$ represent the set of all $n \times n$ real matrices. Given a matrix $S = [s_{ij}] \in M_n(\mathbb{R})$, the digraph of S, denoted by $\Gamma(S)$, is the digraph on n nodes v_i, $i \in \{1, 2, \cdots, n\}$, such that there is an edge in $\Gamma(S)$ from v_j to v_i if and only if $s_{ij} \neq 0$ (cf. [22]).

2.2. Consensus algorithms

Consider vehicles with single-integrator dynamics given by

$$\dot{r}_i = u_i, \quad i = 1, \ldots, n, \tag{2}$$