Self-Adaptive Collective Motion of Swarm Robots

Haitao Zhao, *Senior Member, IEEE*, Hai Liu, *Member, IEEE*, Yiu-Wing Leung, and Xiaowen Chu, *Senior Member, IEEE*

***Abstract*— Collective motion is a fundamental operation of robot swarms by which a group of robots move from a source to a destination in a cohesive way (i.e., connectivity is preserved during these movements). However, the collective motion of robot swarms along preplanned paths has not been well studied. In this paper, we propose self-adaptive collective motion algorithms for swarm robots in 3-D space. Using the proposed collective motion algorithms, robots are able to move along a preplanned path from a source to a destination while satisfying the following requirements: 1) the robots use only one-hop neighbor information; 2) the robots maintain connectivity of the network topology for information exchange; 3) the robots maintain a desired neighboring distance; and 4) the robots are capable of bypassing obstacles without partitioning the robot swarm (i.e., member loss). Our basic idea is to introduce a guidance force and a topology force into the system. The guidance force is used to guide the robots to their destination along the preplanned path. It ensures that the robots continue to move until they reach their destination. The topology force is used to maintain a "good" topology of the robot swarm, such as maintaining connectivity of the network topology and the desired distance between neighboring robots. The resultant of the guidance and topology forces determines the movement of a robot. We develop collective motion algorithms for three cases: 1) no obstacles or leaders; 2) no obstacles with a leader; and 3) with obstacles (with and without a leader). Extensive simulations are conducted to evaluate performance of the proposed algorithms. The simulation results show that: 1) our algorithms meet all the requirements; 2) our algorithms are resistant to GPS errors and robot failures; and 3) self-adaptive control of our algorithms makes network topologies more stable and significantly saves travel time of swarm robots.**

***Note to Practitioners*—We propose self-adaptive collective motion algorithms that enable swarm robots to move along a preplanned path from a source to a destination in 3-D space. Our algorithms use only one-hop neighbor information and operate without central controllers. The algorithms are designed to be self-adaptive in the sense that robots are able to dynamically determine proper moving parameters, based on their environments and statuses. With the proposed algorithms, swarm robots are able to: 1) maintain connectivity and a desired neighboring distance during movement; 2) bypass obstacles without member loss (i.e., the robot swarm is partitioned); and 3) be resistant to GPS errors and robot failures. We address both cases of with a leader and without a leader. Simulation results show effectiveness of our algorithms which can be applied in applications of surveillance, search and rescue, mining, agricultural foraging, autonomous military units, and distributed sensing in micromachinery or human bodies.**

***Index Terms*— Collective motion, mobility control, robotic networks, swarm robots.**

I. Introduction

NEW ROBOTICS applications have been realized using groups of robots that are able to communicate with each other to form a robot swarm. For instance, in October 2016, U.S. military officials in California demonstrated a swarm of 103 microdrones at China Lake. These microdrones were connected via wireless communication and demonstrated advanced swarm behaviors, such as collective decision making, adaptive formation flying, and self-healing. In November 2016, Intel developed a light show incorporating 500 shooting star drones. These drones were equipped with LEDs and able to spell out recognizable words and numbers while hovering.

Swarm robots could be various types of mobile sensors, unmanned vehicles (e.g., Google driverless cars), unmanned aerial vehicles (UAVs), autonomous underwater vehicles/unmanned underwater vehicles (AUVs/UUVs), and unmanned ships. Each robot is usually battery powered and equipped with a wireless transceiver to exchange information with its neighbors within communication range. The robot can carry a variety of electromechanical sensors to explore and interpret their environments. With the capability of moving, sensing, and communicating, swarm robots have great potential to contribute sharing economy in applications including surveillance, search and rescue, mining, agricultural foraging, autonomous military units, and distributed sensing in micromachinery or human bodies. For example, swarm of UAVs can be shared by farmers to perform large-scale aerial crop dusting and be shared by geologists to perform geological prospecting. In disaster rescue missions, swarm robots can be sent into the places that are too dangerous for human workers. These robots can detect life signs via infrared sensors. Moreover, artists have used robotic swarm techniques to realize new forms of interactive art.

Manuscript received October 22, 2017; revised March 19, 2018 and April 25, 2018; accepted May 15, 2018. Date of publication June 22, 2018; date of current version October 4, 2018. This paper was recommended for publication by Associate Editor J. Cheng and Editor M. P. Fanti upon evaluation of the reviewers' comments. This work was supported in part by the Faculty Development Scheme under Project UGC/FDS14/E01/17 and the National Science Foundation of China under Grant 61471376. *(Corresponding author: Haitao Zhao.)*

H. Zhao is with the College of Electronic Science, National University of Defense Technology, Changsha 410073, China (e-mail: haitaozhao@nudt.edu.cn).

H. Liu is with the Department of Computing, Hang Seng Management College, Hong Kong (e-mail: hliu@hsmc.edu.hk).

Y.-W. Leung and X. Chu are with the Department of Computer Science, Hong Kong Baptist University, Hong Kong (e-mail: ywleung@comp.hkbu.edu.hk; chxw@comp.hkbu.edu.hk).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TASE.2018.2840828

TABLE I
DIFFERENCES BETWEEN THE PROPOSED SOLUTION AND EXISTING WORK

	Problems addressed	During movement		Encountering obstacles		Self-adaptive collective motion
		Maintaining desired distance between neighbors	Maintaining connectivity	Avoiding obstacles	Preventing member loss	
Proposed solution	Moving from source to destination along a pre-planned path	Yes	Yes	Yes	Yes	Yes
[20]	Modeling collective motion without leaders	No	No	Yes	No	No
[6], [11], [21], [25], [26], [29]		No	No	No	No	No
[12]	Modeling collective motion with leaders	No	No	Yes	No	Yes
[2], [5]		No	No	Yes	No	No
[19]	Tracking targets	Yes	No	No	No	No
[31]		No	No	No	No	No
[28]		Yes	Yes	Yes	No	No
[32]	Maintaining network connections	Yes	Yes	No	No	No
[10]		Yes	Yes	Yes	Yes	No
[23]		No	Yes	Yes	Yes	No
[17], [22]	Forming topologies and patterns	Yes	Yes	No	No	No
[34]		No	No	No	No	No
[8]	Maximizing coverage area	Yes	Yes	No	No	No
[24]		Yes	No	No	No	No
[13]	Describing vehicle behaviors	No	No	Yes	No	No
[14]		No	No	No	No	No

In robot swarms, collective motion is a fundamental operation in which a group of robots move from a source to a destination in a cohesive way, such that connectivity is preserved during movement. In nature, collective motion is a widely observed phenomenon in insect swarms, bird flocks, and fish schools. There are basically two types of collective motion: motion with and without a leader. In the collective motion with a leader (such as bird flocks), there is normally a single leader; other members of the group are followers who follow the leader's movement. In the collective motion without a leader (such as insect swarms), all of the robots play an equal role and move individually while maintaining contact with each other. The collective motion of robots is important in many practical applications and the three examples are as follows.

1) In search and rescue operations, a group of robots move along a preplanned path while collecting data and searching for targets across a wide area [9], [33].
2) In film production, a set of UAVs can cooperatively photograph or film an event from different angles to achieve information diversity (e.g., 3-D map generation [15]).
3) In military operations [7], a troop of robots (e.g., UAVs/UUVs) may be ordered to move to a specific place. It is necessary for the robots to maintain connectivity during movement, so that they can alert each other to environmental threats and attacks [1], [30].
4) Newly emerged techniques such as mobile cloud and D2D may also require the collective motion of robots or smart devices, which arouses research on energy optimization techniques [36]–[39].

The literature on collective motion can be classified into the following three categories.

1) The work in the first category assumes that there is no leader in some nature creatures such as insect swarms and fish schools [6], [11], [20], [21], [25], [26], [29]. The objective is to model the collective motion of these creatures without leaders.
2) The work in the second category [2], [5], [12] tries to model the collective motion of nature creatures with leaders such as bird flocks.
3) The third category includes the work that applied collective motion to specific applications, such as tracking targets [19], [28], [31], forming specific topologies and patterns [17], [22], [34], maximizing the coverage area [8], [24], describing vehicle behaviors [13], [14], and maintaining network connections [10], [23], [32].

Table I summarizes the differences between the proposed solution and previously reported studies. To the best of our knowledge, the collective motion of swarm robots along a preplanned path has not been studied in detail.

We study the collective motion of swarm robots that move along a preplanned path from a source to a destination in 3-D space. The contribution of this paper is threefold.

1) We propose self-adaptive collective motion algorithms for both swarm robots with and without a leader. The proposed algorithms use only one-hop neighbor information and enable swarm robots to maintain topological connectivity and a desired neighboring distance during movement. The collective motion algorithms enable the

robots to be self-adaptive by dynamically adjusting their movement parameters based on their environments and statuses.

2) We propose an obstacle avoidance algorithm for swarm robots. The proposed algorithm enables the robots to bypass obstacles of any shape and size without partitioning the swarm.
3) We conduct extensive computer simulations to evaluate performance of the proposed algorithms. The results show the following.
   a) Our algorithms enable swarm robots to maintain connectivity and the desired neighboring distance.
   b) Our algorithms are resistant to GPS errors and robot failures.
   c) Self-adaptive control of our algorithms makes network topologies more stable and significantly saves travel time of swarm robots.

The rest of this paper is organized as follows. System models and problem specification are given in Section II. We propose our self-adaptive collective motion algorithms in Section III. Computer simulations are presented in Section IV. We conclude this paper in Section V.

II. System Models and Problem Specification

We consider a 3-D space where there are N robots. Each robot is equipped with a wireless transceiver with a communication range of R_c and is aware of its own position by a GPS device (outdoor environments). In indoor environments, where GPS is not applicable, the robot can adopt indoor localization methods (e.g., via anchor nodes [35]) to compute its position. Two robots are able to detect the presence of each other and exchange information if they are within the communication range of each other. The robot could carry sensors to explore and understand environments. In some applications, a desired distance R_d is required for neighboring robots (e.g., for maximizing sensing and searching coverage of the robots). We have $R_d \leq R_c$; otherwise, the network topology is disconnected. The robot is able to move, and its velocity is adjustable in the range of $[V_{\min}, V_{\max}]$. When $V_{\min} = V_{\max}$, the robot moves with a constant velocity. N robots are initially deployed at the source S and are requested to move from S to a destination T along a preplanned path P which is defined by a sequence of coordinates. That is, $P = \{\langle x_i, y_i, z_i\rangle | i = 1, 2, \ldots, K\}$ where $P_i = \langle x_i, y_i, z_i\rangle$ denotes the ith position of the preplanned path. $P_1 = \langle x_1, y_1, z_1\rangle$ and $P_K = \langle x_K, y_K, z_K\rangle$ are coordinates of S and T, respectively.

The collective motion of swarm robots is to control the motion of the robots to fulfill the following requirements.

1) It prefers localized motion control-based solely on one-hop neighbor information.
2) The robot swarm is required to maintain connectivity during motion, so that the robots are able to communicate with each other and exchange necessary information.
3) The robots are requested to maintain the desired distance R_d from their neighbors during motion.
4) To cope with dynamic environments, the collective motion of robots is expected to be self-adaptive. For example, when the network topology changes suddenly (e.g., due to a sharp bend), the robots will decrease their velocities to maintain connectivity.

The primary objective is to minimize the travel time which is defined as the time it takes to move from source S to destination T. (In many life-critical and military applications, swarm robots are requested to move from the source to the destination as soon as possible.) The second objective is to minimize total travel distances of the robots so as to save their energy.

III. Collective Motion Algorithms

In this section, we propose three localized and self-adaptive collective motion algorithms for the problem defined in Section II. The three algorithms are designed for the following three cases of robot swarms: 1) no obstacles or leaders; 2) no obstacles with a leader; and 3) with obstacles (with and without a leader).

A. No Obstacles or Leaders

Our basic idea is to introduce a guidance force and a topology force into the system. The guidance force, denoted by vector $\overrightarrow{F_G}$, is to guide the robots to the destination along the preplanned path. It ensures that the robots continue to move until they reach the destination. The topology force, denoted by vector $\overrightarrow{F_T}$, is to maintain a "good" topology of the robot swarm, in which network connectivity and the desired distance between neighboring robots are maintained. The guidance force and the topology force are applied to each robot, and the robot moves as directed by the resultant force.

Note that the preplanned path P consists of a sequence of coordinates P_i, $i = 1, 2, \ldots, K$, which is known to each robot. During the movement, a robot starts with P_1 (source S) and moves towards P_{i+1} after it visits P_i, $i = 1, 2, \ldots, K-1$. The visits continue sequentially until the robot reaches P_K (destination T). Suppose that the robot has visited P_i and the next coordinate is P_{i+1}. Let $C = \langle x_c, y_c, z_c\rangle$ denote the current position of the robot. The guidance force acting on the robot is defined as follows:

$$\overrightarrow{F_G} = \overrightarrow{\sqrt{(x_{i+1}-x_c)^2 + (y_{i+1}-y_c)^2 + (z_{i+1}-z_c)^2}} \quad (1)$$

where the magnitude of $\overrightarrow{F_G}$ is the Euclidean distance between C and P_{i+1}, and the direction of $\overrightarrow{F_G}$ is from C to P_{i+1}, as shown in Fig. 1. From this definition, it is evident that the magnitude of the guidance force is greatest at the start of each movement and gradually decreases as the robot moves from P_i to P_{i+1}, $i = 1, 2, \ldots, K-1$. This is necessary, as the robots are expected to slow down as they draw nearer to each intermediate stop of P, so that a good and stable network topology can be formed before the next move.

To implement the topology force, we adopt the virtual force models presented in our preliminary work [27]. We assume that there are attractive and repulsive virtual forces between two neighboring robots. However, only one of these virtual forces acts at a given time. Specifically, let $D(i, j)$,

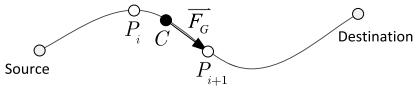


Fig. 1. Illustration of guidance force $\overrightarrow{F_G}$.

$0 \le D(i, j) \le R_c$, denote distance between two neighboring robots i and j. The attractive force takes effect if $R_d < D(i, j) \le R_c$; otherwise, the repulsive force takes effect ($D(i, j) \le R_d \le R_c$). Thus, two neighboring robots are attracted if their distance is greater than the desired neighboring distance (to maintain network connectivity) and repelled otherwise (to maintain the desired distance). Let $\overrightarrow{f_A(i, j)}$ and $\overrightarrow{f_R(i, j)}$ denote the attractive force and the repulsive force, respectively, acting on robot i that originate from its neighbor j. They are defined as follows:

$$\overrightarrow{f_A(i, j)} = \overrightarrow{\sqrt{(x_i - x_j)^2 + (y_i - y_j)^2 + (z_i - z_j)^2}} \tag{2}$$

$$\overrightarrow{f_R(i, j)} = \begin{cases} \overrightarrow{M/D(i, j)^2 - M/R_d^2}, & \text{if } 0 < D(i, j) \le R_d \\ \overrightarrow{0}, & \text{if } D(i, j) > R_d \end{cases} \tag{3}$$

where M is a positive constant. The topology force of a robot, say i, is the resultant force of all of the attractive and repulsive forces acting on the robot from its neighbors, denoted by $N(i)$. This can be expressed as follows:

$$\overrightarrow{F_T} = \alpha_A \sum_{j \in N(i)} \overrightarrow{f_A(i, j)} + \alpha_R \sum_{j \in N(i)} \overrightarrow{f_R(i, j)} \tag{4}$$

where coefficients α_A and α_R, $0 \le \alpha_A, \alpha_R \le 1$ and $\alpha_A + \alpha_R = 1$, are the weights of the attractive and repulsive forces, respectively.

Once $\overrightarrow{F_G}$ and $\overrightarrow{F_T}$ have been determined, the resultant force $\overrightarrow{F_{\mathrm{RS}}}$ can be modeled as follows:

$$\overrightarrow{F_{\mathrm{RS}}} = \beta_G \overrightarrow{F_G} + \beta_T \overrightarrow{F_T} \tag{5}$$

where coefficients β_G and β_T are the weights of the guidance and topology forces, respectively, $0 \le \beta_G, \beta_T \le 1$, and $\beta_G + \beta_T = 1$. Each robot simply follows the above resultant force to move.

Collective motion is carried out in multiple rounds. In each round, each robot computes the resultant force $\overrightarrow{F_{\mathrm{RS}}}$ acting on itself according to (1)–(5), and moves along the direction of $\overrightarrow{F_{\mathrm{RS}}}$. Velocity of the robot is determined by the magnitude of $\overrightarrow{F_{\mathrm{RS}}}$. The velocity becomes larger if $\overrightarrow{F_{\mathrm{RS}}}$ has larger magnitude and vice versa. Since the magnitude of $\overrightarrow{F_{\mathrm{RS}}}$ varies from 0 to $+\infty$, it is necessary to map this magnitude to a finite velocity from 0 to $V_{\max}$, where $V_{\max}$ is the maximum velocity of robots ($V_{\min} = 0$ in this case). To perform this mapping, we select the trigonometric function arctan() such that the velocity of robot i, denoted by v_i, is given by

$$v_i = \arctan(|\overrightarrow{F_{\mathrm{RS}}}|) \times (2/\pi) \times V_{\max}. \tag{6}$$

The above nonlinear mapping is suitable based on the following reasons. First, the mapping function is an increasing function. Velocity increases with the increase of the magnitude of $\overrightarrow{F_{\mathrm{RS}}}$. Second, when the force has larger magnitude, the velocity is less sensitive to any further increase in the force magnitude.

Self-adaptive collective motion requires robots to dynamically determine proper ratios of α_A/α_R and β_G/β_T, depending on their environments and statuses. Specifically, if a robot finds that its average neighboring distance is much smaller than the desired distance R_d, it decreases the ratio of α_A/α_R, which would increase the effect of the repulsive force acting on it. If its average neighboring distance is much larger than R_d, the robot increases the ratio of α_A/α_R so as to increase the effect of the attractive force acting on it. This mechanism prevents the robot from being disconnected from the swarm while keeping the desired neighboring distance R_d. If a robot has very stable connections to its neighbors and is reliably maintaining the desired distance from them, it can increase the ratio of β_G/β_T to increase the effect of the guidance force and thereby increase its velocity. If a robot detects distortion of its neighbor topology (e.g., large diversity of the neighboring distances), it can decrease the ratio of β_G/β_T to increase the effect of the topology force. This mechanism enables the swarm robots to move at high velocities while maintaining a stable network topology.

The detailed collective motion algorithm is given in Table II. In the algorithm, Avr(i) and SD(i), respectively, denote the average neighboring distance of robot i and the standard deviation of neighboring distances of robot i.

B. No Obstacles With a Leader

In Section III-A, we assume that there is no leader in robot swarms. In this case, all of the robots are aware of the preplanned path P and move individually. In some applications (such as military missions), however, P may be dynamically updated and frequently changed to avoid information leaks. In practice, updating P in all of the robots incurs a considerable cost because of the large population size of the swarm. It is often more efficient to specify a robot as the leader and have other robots follow it. In this scenario, only the leader needs to be notified if P is updated.

In this section, we assume that one robot in the swarm is the leader and the other robots are followers. This phenomenon is frequently observed in bird flocks. The leader is the only robot that knows the preplanned path P. The follower robots move by following the leader. Note that there are usually a large number (e.g., hundreds or thousands) of robots in a swarm. It is impractical to connect all of the followers to the leader directly because the communication range is limited. Hence, some follower robots cannot directly contact the leader robot and they have no knowledge of P. The collective motion algorithm for this problem must be carefully designed to handle this lack of information.

Nagy *et al.* [18] investigated the homing flights of pigeons. They found that a hierarchical organization of group flight could be more efficient than an egalitarian organization. Inspired by this paper, we propose to incorporate a hierarchical organization into the collective motion algorithm for swarms

TABLE II
COLLECTIVE MOTION ALGORITHM (WITH NO OBSTACLES OR LEADERS)

//Initialization:
P: the pre-planned path;
V_{max}: the maximal velocity of robots;
Th_1-Th_5: predetermined thresholds;
Δ_1, Δ_2: stride lengths of adjustment;

//For each robot *i*:
//Thread A: Information broadcast
While (TRUE) **do**
Broadcast HELLO to 1-hop neighbors, which encodes its current position;
End while
//Thread B: Information update
While (Receive a HELLO packet from j) **do**
If ($j \notin N(i)$)
Add j to neighbor set $N(i)$ and record its position;
Else
Update position of j;
End if
End while
//Thread C: Motion control
While (not reach the destination yet) **do**
//Self-adaptive mechanism
If ($Avr(i) - R_d > Th_1$) *//neighbors are too far*
Increase ratio α_A/α_R by Δ_1;
End if
If ($R_d - Avr(i) > Th_2$) *//neighbors are too close*
Decrease ratio α_A/α_R by Δ_1;
End if
If ($SD(i) \le Th_3$ and $R_c - Avr(i) > Th_4$)
//topology is stable and neighbors are strongly connected
Increase ratio β_G/β_T by Δ_2;
End if
If ($SD(i) > Th_5$) *//topology is not stable*
Decrease ratio β_G/β_T by Δ_2;
End if
Compute resultant force $\overrightarrow{F_{RS}}$ according to (5);
Compute velocity v_i according to (6);
Move along the direction of $\overrightarrow{F_{RS}}$ with a velocity of v_i;
End while

with a leader. Specifically, each robot, say i, is associated with a hierarchical index, denoted by l_i, which indicates its closeness to the leader robot. The hierarchical index of the leader is set to be 0. All of the followers that are directly connected to the leader have a hierarchical index of 1. For a single robot i, if the lowest hierarchical index of its neighbors is w, then $l_i = w + 1$. In fact, the hierarchical index of a robot is the length of the shortest path, in terms of number of hops, from the robot to the leader robot. Fig. 2 illustrates the hierarchical organization of a robot swarm.

Based on the hierarchical organization, we redesign models of the guidance and topology forces as follows.

1) Leader: The leader is the only robot that is aware of the preplanned path P. Therefore, the guidance force is still valid. The topology force acting on the leader comes from

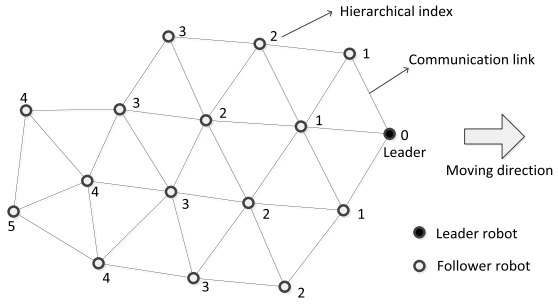


Fig. 2. Illustration of hierarchical organization.

all its direct neighbors (those with a hierarchical index of 1). Hence, the leader adopts the same models described in (1)–(5) to compute the resultant force $\overrightarrow{F_{\mathrm{RS}}}$ and follows its direction to move.

2) Follower: Note that followers have no knowledge of P. Therefore, there is no guidance force acting on the followers. Instead, each follower robot is acted upon by the attractive forces originating from its neighbors with lower hierarchical indices. This is because the neighbors with lower hierarchical indices are closer to the leader and thus have higher probabilities on the right track moving toward the destination. To avoid the collision of robots, the repulsive force takes effect when two neighboring robots are getting too close. Specifically, each follower, say i, determines the resultant force acting on itself as follows:

$$\overrightarrow{F_{\mathrm{RS}}} = \overrightarrow{F_T} = \alpha_A \sum_{j \in N(i), l_j < l_i} \overrightarrow{f_A(i,j)} + \alpha_R \sum_{j \in N(i)} \overrightarrow{f_R(i,j)} \quad (7)$$

where the attractive forces originate from its neighbors with lower hierarchical indices and the repulsive forces originate from all of its neighbors. To be self-adaptive, the followers should dynamically determine a proper ratio of α_A/α_R during their movements. The process is the same to that described in Section III-A.

Since the leader adopts the same collective motion algorithm in Table II, we present the collective motion algorithm for followers in Table III.

C. With Obstacles

In practical applications, there usually exist obstacles of various shapes and sizes. Robot swarms must be capable of avoiding these obstacles without member loss (i.e., partition of the swarm). To address this problem, we develop an obstacle avoidance algorithm for collective motion of swarm robots. We assume that the robots are able to detect and sense real-world obstacles by either ultrasound or vision techniques. This assumption is valid because obstacle detection is already available in off-the-shelf robot products. For example, the DJI Phantom 4 drone (https://www.dji.com/phantom-4) is equipped with five cameras which capture video from various angles. The images captured by these cameras are analyzed by computer vision software that constructs a 3-D model of the drone's environment, allowing it to detect obstacles.

TABLE III
COLLECTIVE MOTION ALGORITHM FOR FOLLOWERS (NO OBSTACLES WITH A LEADER)

//Initialization:
V_{max}: the maximal velocity of robots;
Th_1, Th_2: predetermined thresholds;
Δ_1: stride length of adjustment;

//for each follower robot *i*
//Thread A: Information broadcast
While (TRUE) **do**
Broadcast HELLO to 1-hop neighbors, which encodes its hierarchical index and current position;
End while
//Thread B: Information update
While (Receive a HELLO packet from *j*) **do**
If ($j \notin N(i)$)
Add *j* to neighbor set *N*(*i*) and record its position and index;
Else
Update position and index of *j*;
End if
Update its own hierarchical index accordingly;
End while
//Thread C: Collective motion
While (not reach the destination yet) **do**
//Self-adaptive mechanism
If ($Avr(i) - R_d > Th_1$) *//neighbors are too far*
Increase ratio α_A/α_R by Δ_1;
End if
If ($R_d - Avr(i) > Th_2$) *//neighbors are too close*
Decrease ratio α_A/α_R by Δ_1;
End if
Compute resultant force $\overrightarrow{F_{RS}}$ according to (7);
Compute velocity v_i according to (6);
Move along direction of $\overrightarrow{F_{RS}}$ with a velocity of v_i;
End while

We consider robot swarms with and without a leader.

1) Obstacle Avoidance With a Leader: Before encountering an obstacle, swarm robots move by executing the collective motion algorithm presented in Section III-B. When a robot detects an obstacle, it acts as follows, depending on whether it is a follower or the leader.

If the robot is a follower, the resultant force $\overrightarrow{F_{\mathrm{RS}}}$ acting on it is calculated using (6). $\overrightarrow{F_{\mathrm{RS}}}$ can be resolved into two components $\overrightarrow{F_{\mathrm{RS}}^1}$ and $\overrightarrow{F_{\mathrm{RS}}^2}$, as shown in Fig. 3(a). We assume that the obstacle applies an equal and opposite reaction to any action from the robot, in accordance with Newton's third law of motions. Specifically, we assume that there is a virtual support force from the obstacle that cancels out $\overrightarrow{F_{\mathrm{RS}}^2}$ each other. As such, the resultant force on the follower robot is equal to $\overrightarrow{F_{\mathrm{RS}}^1}$.

If the robot is the leader, its responsibility is to lead all of the followers to bypass the obstacle. We propose to use the path planning method presented in [16]. Specifically, when the leader encounters an obstacle at position U, it moves along the boundary of the obstacle using either the left-hand rule or the right-hand rule [16]. That is, either the left hand or the right hand of the leader maintains contact with the obstacle until it reaches point V on the line $P_i P_{i+1}$. The left-hand rule is illustrated in Fig. 3(b). The leader resumes its regular movement at point V. In practice, P_{i+1} may be located inside of the obstacle. In this case, the leader will revisit U after moving along the boundary of the obstacle. At this point, the leader can skip P_{i+1} and take P_{i+2} as the next point along its path, i.e., moving from P_i to P_{i+2}.

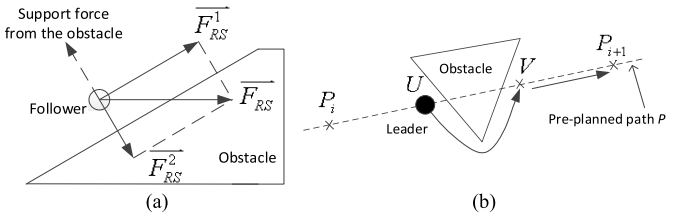


Fig. 3. Obstacle avoidance of swarm robots with a leader. (a) Follower encounters an obstacle. (b) Leader encounters an obstacle.

2) Obstacle Avoidance Without a Leader: In a robot swarm without a leader, each robot makes its own decisions when encountering an obstacle. As a result, the robots may bypass the obstacle in different directions which consequently results in partition of the swarm (i.e., causing member loss). To tackle the problem, we transform the case of without a leader to the case of with a leader. Our basic idea is to select a temporary leader by using leader/head election algorithms for ad hoc networks (see [3]). Specifically, when a robot detects an obstacle, it proclaims itself as the temporary leader by broadcasting a claim message if it has not received a similar claim from its one-hop neighbors. Any robot that receives a claim message acts as a follower. If multiple robots broadcast the leadership claim simultaneously, the robot with the largest ID is selected as the leader. Once the leader has been determined, it leads all of the robots until they have bypassed the obstacle, at which point the swarm resumes its original movement (i.e., moving without a leader). The detailed algorithm is given in Table IV.

IV. SIMULATION EXPERIMENTS

A. Experimental Setup and Performance Metrics

We use MATLAB 2015b with GUI to implement the proposed algorithms as a simulator which graphically shows the dynamics of the movement process. Readers can download the simulator from [40] to check the movement process. The following metrics are used to evaluate performance of the collective motion algorithms.

1) *Travel Time (T_{travel}):* It is defined as the average time taken by all robots moving from the source to the destination. It indicates how fast the swarm robots can complete collective motion by executing the proposed algorithms.
2) *Travel Distance (L_{travel}):* It is defined as the average distance that all robots have travelled throughout the collective motion process. It indicates how long the swarm robots travel to complete the collective motion.

TABLE IV
OBSTACLE AVOIDANCE ALGORITHM (WITH AND WITHOUT A LEADER)

//Algorithm 4.1 for leader (With a leader):
If (Encounter an obstacle at position U)
  //Refer to Fig. 3(b)
  While (not reach position V yet) **do**
    Move along boundary of the obstacle using the left-hand/right-hand rule [16];
  End while
  If (U=V) *//P_{i+1} is located inside of the obstacle*
    Set next-hop destination P_{i+2};
  End if
End if
Run collective motion algorithm in Table II;

//Algorithm 4.2 for followers (With a leader):
If (Encounter an obstacle)
  //Refer to Fig. 3(a)
  Compute resultant force $\overrightarrow{F_{RS}^{1}}$ according to Fig. 3(a);
  Compute velocity v_i according to (6);
  Move along direction of $\overrightarrow{F_{RS}^{1}}$ with a velocity of v_i;
Else
  Run collective motion algorithm in Table III;
End if

//Algorithm 4.3 (Without a leader):
If (Encounter an obstacle)
  Select a temporary leader by leader election algorithms (e.g., [3]);
End if
While (not bypass the obstacle yet) **do**
  If (I am the temporary leader)
    Run Algorithm 4.1 for leader;
  Else
    Run Algorithm 4.2 for followers;
  End if
End while
Run collective motion algorithm in Table II;

3) *Travel Efficiency (E_{ff}):* It is defined as the ratio of length of the preplanned path to the actual travel distance (L_{travel}). It is evident that a bigger E_{ff} indicates more efficient move of swarm robots.
4) *Travel Variance Index (L_{var}):* It is defined by Jain's fairness index

$$L_{\mathrm{var}} = \left(\sum\nolimits_{i=1}^{N} L_{\mathrm{travel}}^{i}\right)^{2} / N \cdot \sum\nolimits_{i=1}^{N} L_{\mathrm{travel}^{i}}^{2} \qquad (8)$$

where L_{travel}^{i} is the travel distance of robot i throughout the collective motion process. Note that value of L_{var} varies from $1/N$ (the worst case) to 1 (the best case), and is maximized when travel distances of all robots are the same.

1) *Topology Stability (S_{ta})*: It is defined as

$$S_{\mathrm{ta}} = \underset{i=1,2,\ldots,N}{\mathrm{Mean}} \left\{ Q - \sum\nolimits_{j=1}^{Q} u_j^i / Q \right\} \qquad (9)$$

where Q is the total number of iterations in the collective motion and u_j^i, $i = 1, 2, \ldots, N$ and $j = 1, 2, \ldots, Q$, is an indicator function. u_j^i is equal to 0 if robot i's neighbor set $N(i)$ is not changed comparing to that in the last iteration and is equal to 1 otherwise. Note that $S_{\mathrm{ta}} \in [0, 1]$ and a bigger S_{ta} indicates a more stable topology.

In our algorithms, we assume that a HELLO message is broadcasted once a second, which is the same as adopted in many routing protocols such as AODV and DSR. Other simulation parameters are given in Table V.

TABLE V
PARAMETERS USED IN THE EXPERIMENTS

Parameter	Value
Number of robots (N)	5 ~ 30
Communication range (R_c)	300m
Desired range (R_d)	260m
Maximal velocity (V_{max})	20m/s
Types of the pre-planned path	Straight line, sine curve, and circle
Length of the pre-planned path	10km
Obstacle	A circle with diameter 600m

B. Examples

We present four examples in Figs. 4–7 to show dynamics of swarm robots by executing the proposed collective motion algorithms. For easy illustration, the first three examples (Figs. 4–6) are conducted in 2-D space and show dynamics of swarm robots in three examples of without a leader, with a leader, and obstacle avoidance, respectively. Fig. 7 shows an example in 3-D space. In Figs. 4–7, the source and the destination are marked by red square and red star, respectively. The dashed line between the source and the destination indicates the preplanned path. Swarm robots are randomly deployed around the source and are initialized as a biconnected network by using the algorithm in [27]. For a robot, three shapes are used to differentiate its roles.

1) Hollow circle denotes the robot in the swarm without a leader.
2) Black-filled circle denotes the leader robot.
3) Gray-filled circle denotes a follower robot.

If the Euclidean distance of two robots is smaller than R_c, there is an edge between them and the two robots can communicate with each other directly. Next, we discuss the three examples.

1) Collective Motion Without a Leader (Fig. 4): Suppose there are 20 robots in a swarm as illustrated in Fig. 4, and the preplanned path is a sine curve. The randomly deployed swarm robots are shown in Fig. 4(a). The initialized biconnected network is shown in Fig. 4(b). Then, the swarm robots collectively move along the path, being controlled by the proposed algorithm in Table II. In the process of collective motion, with the effects of topology force and guidance force, the network keeps a very stable topology and at the same time smoothly moves along the predefined path [Fig. 4(c)–(e)]. Finally, the network arrived at the destination and stopped there [Fig. 4(f)].

2) Collective Motion With a Leader (Fig. 5): In Fig. 5, there are 20 swarm robots randomly deployed and there is a leader located in the center [Fig. 5(a)]. The preplanned

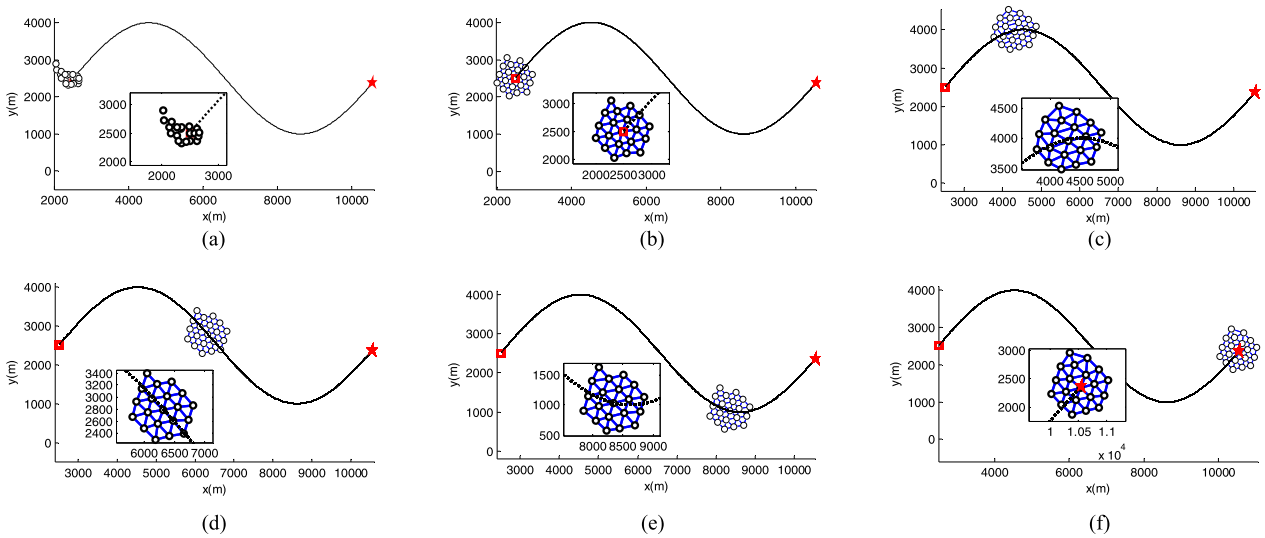


Fig. 4. Dynamics of swarm robots without a leader. ($N = 20$, the preplanned path is a sine curve.) (a) Randomly deployed swarm robots. (b) Initialized network topology ($t = 0$ s). (c) Moving along the preplanned path ($t = 200$ s). (d) Moving along the preplanned path ($t = 400$ s). (e) Moving along the preplanned path ($t = 600$ s). (f) Swarm robots arrive at the destination (final state, $t = 835$ s).

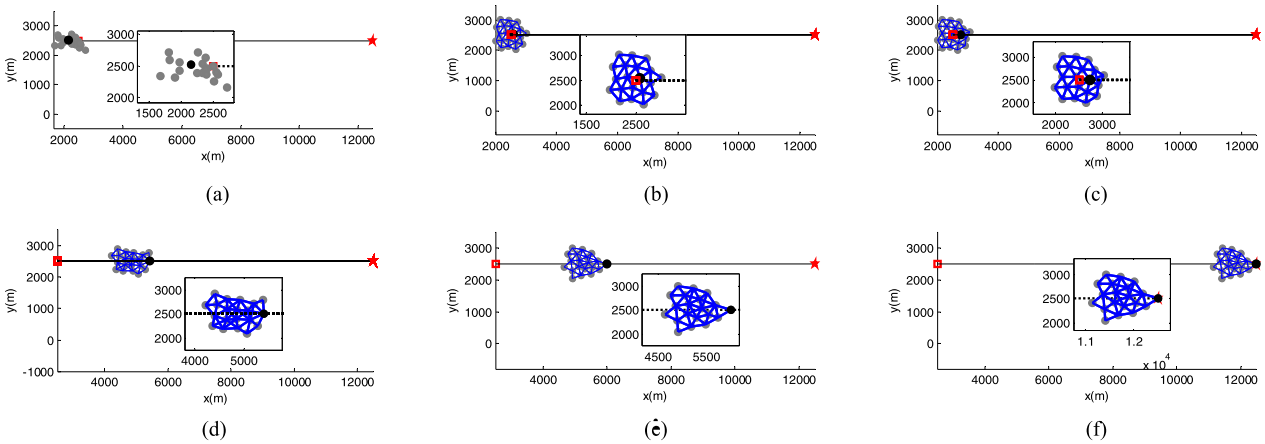


Fig. 5. Dynamics of swarm robots with a leader. ($N = 20$, the preplanned path is a straight line.) (a) Randomly deployed swarm robots. (b) Initialized network topology ($t = 0$ s). (c) Moving along the preplanned path ($t = 20$ s). (d) Moving along the preplanned path ($t = 100$ s). (e) Moving along the preplanned path ($t = 145$ s). (f) Swarm robots arrive at the destination (final state, $t = 665$ s).

path is a straight line. After the robot swarm is initialized to be biconnected [Fig. 5(b)], it starts the collective motion by executing the proposed algorithm in Table III. As the swarm moves, Fig. 5(b)–(d) shows that the leader is moving to the front of the swarm gradually, since only it is affected by the guidance force. In this process, the network topology is changing. However, the topology will become stable when the leader comes to the very front as shown in Fig. 5(e). This stable network topology will keep until to the destination [Fig. 5(f)]. Note that under this scenario, the network topology at the destination is not the same as at the source, but it still keeps the following two virtues: 1) the network is biconnected and 2) the distances between two neighbors are all kept as the desired value.

3) Obstacle Avoidance (Fig. 6): Now, we present an example to show the dynamics of the collective motion process in avoiding an obstacle. For clearance, we set seven robots in the network and an obstacle with a diameter of 600 m appears on the preplanned path (see Fig. 6). Before encountering the obstacle, the network is under the without-leader scenario, and all robots are performing collective motion controlled by the algorithm in Table II [Fig. 6(a)]. Fig. 6(b) shows that when the first robot meets the obstacle, it claims itself as the temporary leader. At the same time, it broadcasts this change to its one-hop neighbors via HELLO packets. And via hop-by-hop broadcasts, all the robots in the network know that there is an obstacle and a temporary leader, and the network changes to the with-leader scenario, i.e., controlled by the algorithm in Table III [Fig. 6(c)]. Thereafter, the network uses left-hand rule to avoid the obstacle [Fig. 6(d)]. And after bypassing the obstacle, the network changes to collectively move under the without-leader scenario as before encountering the obstacle.

4) Example in 3-D Space (Fig. 7): The proposed algorithms are applicable to 3-D space. Due to display difficulty of 3-D, we present the example where the preplanned path is a straight line and there is no leader and no obstacle. The dotted lines

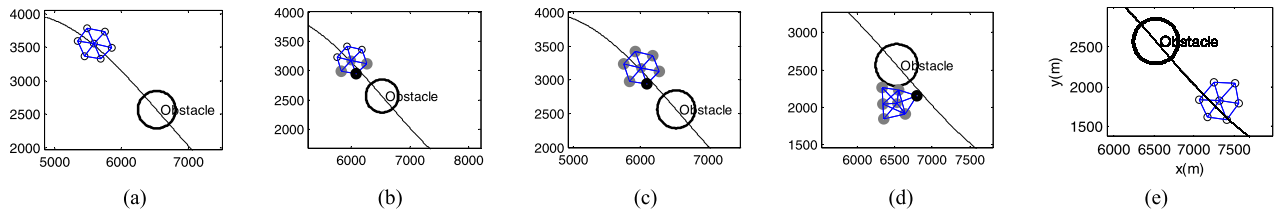


Fig. 6. Dynamics of swarm robots bypassing an obstacle. (Without a leader, $N = 7$, the obstacle is a circle with diameter 600 m.) (a) Before encountering an obstacle. (b) Encounter the obstacle. (c) Switch to the with-a-leader case. (d) Bypass the obstacle. (e) Resume the original movement (i.e., moving with a leader).

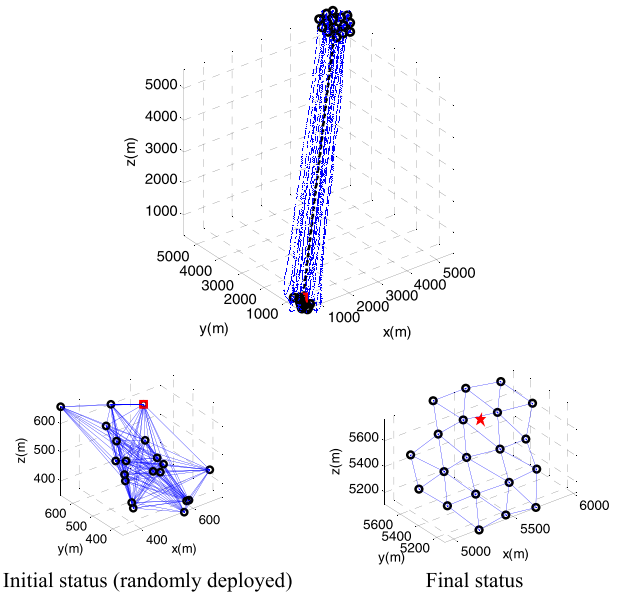


Fig. 7. Collective motion without a leader in 3-D space. (a) Initial status (randomly deployed). (b) Final status.

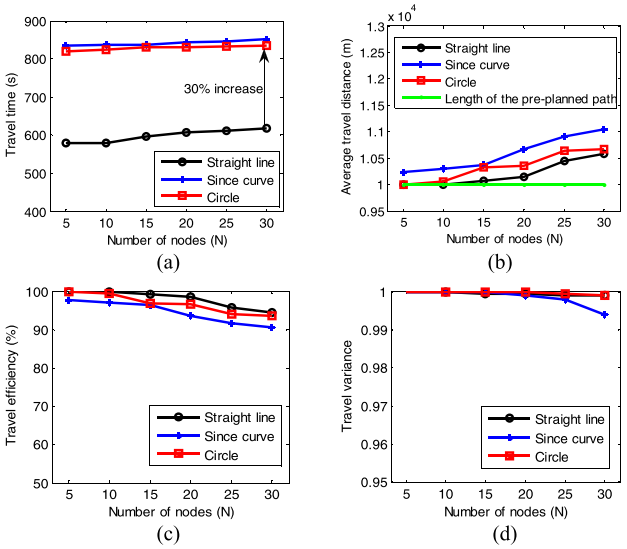


Fig. 8. Collective motion without a leader. (a) Travel time. (b) Average travel distance. (c) Travel efficiency. (d) Travel variance index.

in Fig. 7 represent moving traces of swarm robots. It is shown that swarm robots start with a random topology and end with a stable topology at the destination.

C. Performance Evaluation

We first investigate collective motion of swarm robots without and with a leader. Then, we evaluate performance of the obstacle avoidance algorithm proposed in Section III-C. We further study resistance to GPS errors. Last, we investigate the self-adaptive behaviors of swarm robots. The metrics presented in Section IV-A are used to evaluate the proposed algorithms. Each result presented in Figs. 8, 9, 11, 12, and 14 is the average value of 100 separate runs.

1) Collective Motion Without a Leader: We set three representative kinds of preplanned path: straight line, a cycle of sine curve (the amplitude is 1500 m) and a circle in the simulation. The total length of the preplanned path is fixed to 10 km for fair comparison. Fig. 8 shows the simulation results for the metrics of travel time, travel distance, travel variance, and travel efficiency. Note that the network is always stable ($S_{\text{ta}} = 1$) in all simulation settings under this scenario, and therefore not plotted here.

From Fig. 8(a), we can see that the network uses the least time to complete the collective motion when the preplanned path is a straight line, and the travel time may be consumed more that 30% when the preplanned path is a circle or a sine curve. Another interesting observation is that the travel time is only slightly increases with the number of robots in the network, which is resulted from the fact that under this scenario, all robots have the information of the preplanned path. However, Fig. 8(b) shows that the average travel distance of the robots apparently increases with the number of robots. This is because more robots in the network means more motion interaction with neighbors to keep the network topology, and hence accumulates to a longer travel distance. It is also noticed that with the same settings, robots travel most when the path is sine curve and least when the path is straight line. But in general, the travel distance is very close to the length of the preplanned path, which can also be show in the travel efficiency that plotted in Fig. 8(c). It shows that although the travel efficiency decreases with the robot number, it is always more than 90%. Fig. 8(d) shows that under all settings, the travel variance index is very close to 1 (more than 0.99), which means that all the robots move almost the same distance throughout the collective motion. In other words, the control algorithm is fair to all robots, which is helpful to guarantee

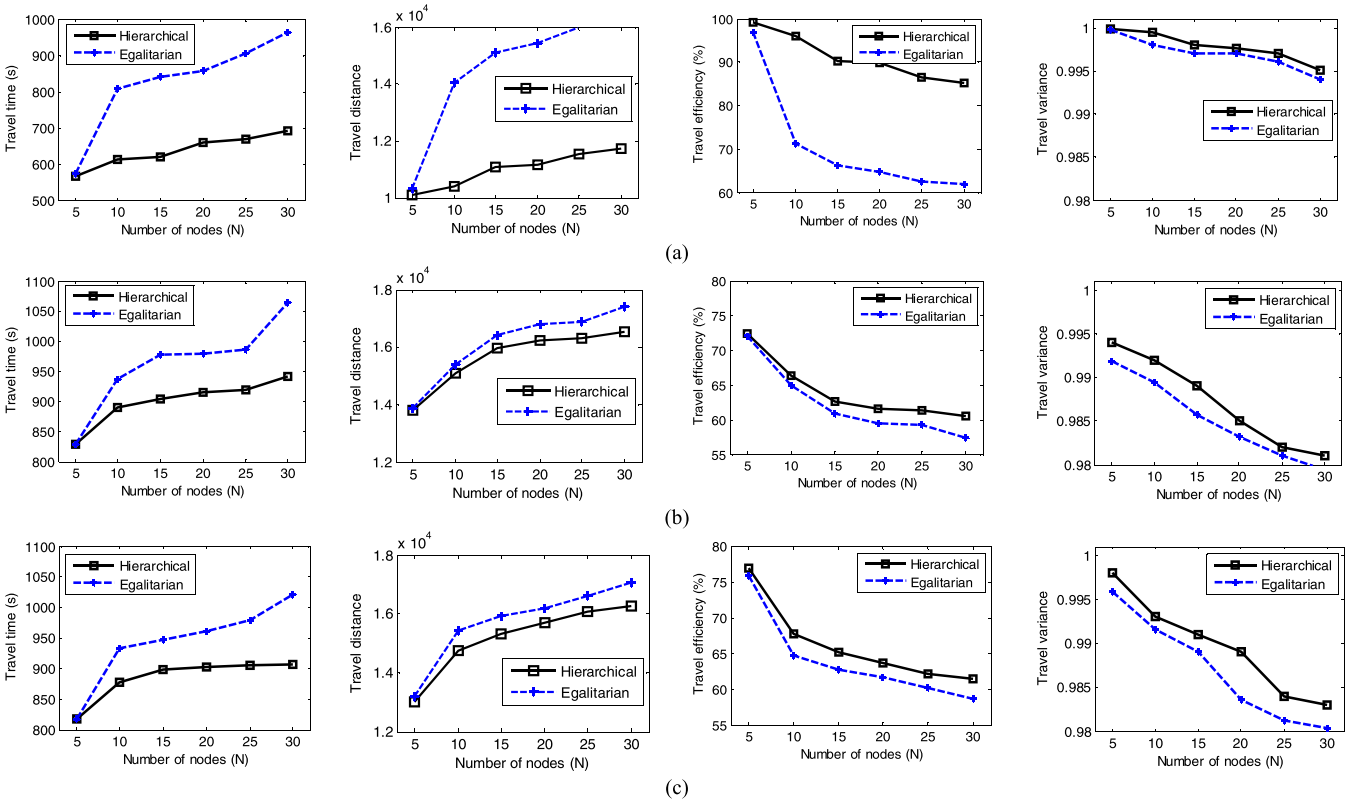


Fig. 9. Collective motion with a leader. (a) Preplanned path is a line. (b) Preplanned path is a curve. (c) Preplanned path is a circle.

that all the robots consume the same energy in the collective motion, prolonging the network lifetime.

2) Collective Motion With a Leader: Using the same simulation settings, we evaluate the performance of the collective motion under the with-leader scenario. We compare the proposed hierarchical strategy with the egalitarian strategy that used in [17] and [18].

We can see that whatever type of the preplanned path is, the hierarchical strategy always outperforms the egalitarian strategy in the following aspects: 1) It shortens the travel time and the travel distance and 2) It increases the travel efficiency and the travel variance index. In other words, the hierarchical strategy can make the network complete the motion quicker and also in a more efficient way. The difference between the hierarchical strategy and the egalitarian strategy is more notable when the preplanned path is straight line, as illustrated in Fig. 9(a). It shows that when there are more than 10 robots, the robot in hierarchical strategy can save more than 200 s in travel time and 3000 m in travel distance, and also the efficiency can be increased by more than 20%. Nevertheless, when the robot number is small (for instance, $N = 5$), the advantage of hierarchical strategy is not so significant. This is because when the robot number is small, most robots are in the one-hop range to the leader and therefore have the same hierarchical priority. Therefore, the hierarchical strategy works almost the same as the egalitarian strategy.

3) Obstacle Avoidance: In this section, we evaluate the collective motion performance in the process of obstacle avoidance, which starts from the moment that at least one robot encounters an obstacle and ends when all robots bypass it. We use the following metric to measure how efficient does a robot swarm avoid an obstacle:

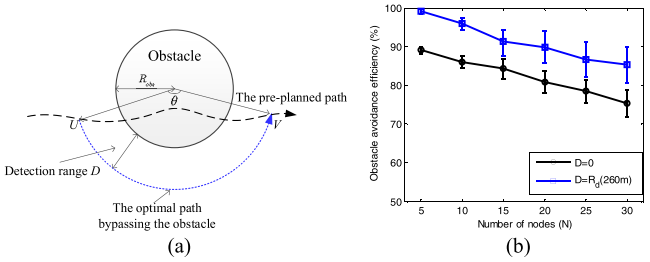


Fig. 10. Obstacle avoidance. (a) Illustration of the optimal path for obstacle avoidance. (Left-hand rule is applied.) (b) Efficiency.

$$E_{\text{oa}} = L_{\text{opt}}/L \tag{10}$$

where L is the average travel distance in the obstacle avoidance process, and L_{opt} is the shortest distance to bypass the obstacle. Clearly, we have $E_{\text{oa}} \in (0, 1]$. The obstacle avoidance efficiency is maximized when $E_{\text{oa}} = 1$ (i.e., $L = L_{\text{opt}}$).

For simplicity, we assume that the obstacle is a circle with a radius of R_{obs} in the simulation. Let D denote detection range of robots. Let θ be the angle between points U and V on the circle with radius of $R_{\text{obs}} + D$, where U and V are, respectively, the start and end positions of bypassing the

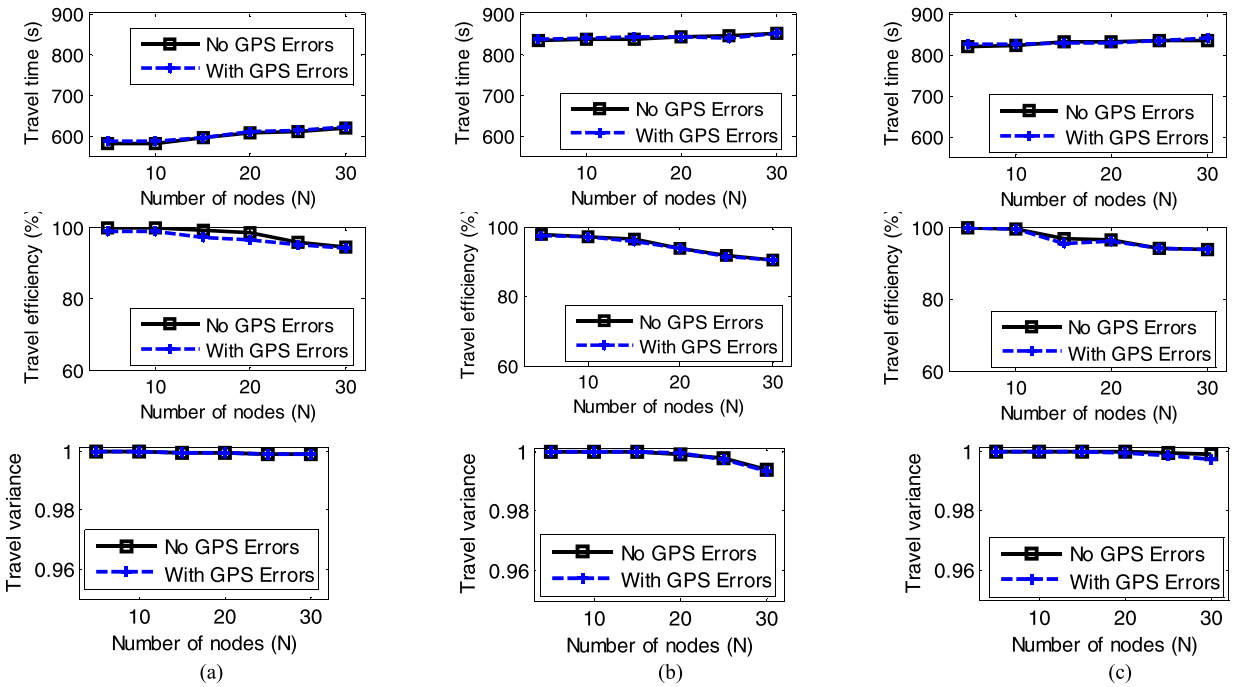


Fig. 11. Resistant to GPS errors (without a leader). (a) Line. (b) Curve. (c) Circle.

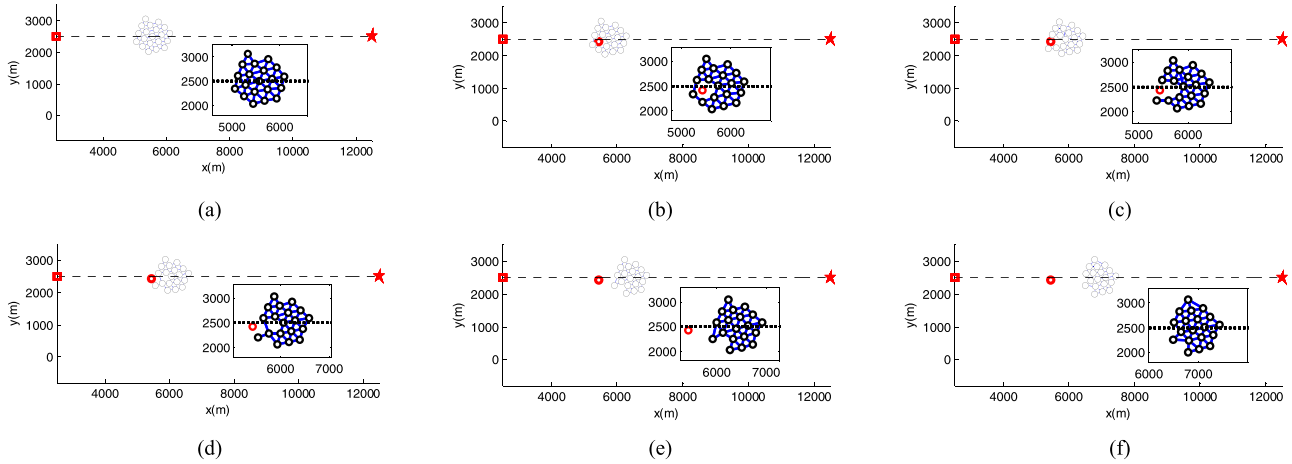


Fig. 12. Resistant to robot failures (without a leader). (a) All robots work properly ($t < 200$ s). (b) One robot fails ($t = 200$ s). (c) Swarm robots are self-healing ($t = 210$ s). (d) Swarm robots are self-healing ($t = 220$ s). (e) Swarm robots keep moving without the failure robot ($t = 230$ s). (f) Swarm robots form a new and stable topology ($t > 240$ s).

obstacle [see Fig. 10(a)]. In this case, we have

$$E_{\text{oa}} = \frac{L_{\text{opt}}}{L} = \frac{2\pi(R_{\text{obs}} + D) \times (\theta/2\pi)}{L} = \frac{\theta(R_{\text{obs}} + D)}{L}. \tag{11}$$

Fig. 10(b) shows the simulation results on obstacle avoiding efficiency. It shows that the efficiency drops with the number of robots, which is in accordance with the simulation before, and the efficiency increases with the detection range. It is reasonable in that if the detection range is large enough, avoiding obstacle would be just as collective moving along part of a circle. And when $D = R_d$, the efficiency can be more than 85% when $N < 30$.

Interestingly, comparing the results with $D = R_d$ in Fig. 10(b) and the second plot in Fig. 9(c), we can find that the obstacle avoiding efficiency is even better than the travel efficiency when moving along a circle under the with-leader scenario. This is because Fig. 9 includes a warm-up process, in which the leader moves to the front of the network, and also a stopping process, in which the network iteratively stops around the destination. And these two processes will decrease the whole travel efficiency.

4) Resistant to GPS Errors and Robot Failures: According to the U.S. government information on GPS and real-world data from the Federal Aviation Administration, standard civilian GPS receivers can provide better than 3.5-m horizontal accuracy. Therefore, to simulate the effect of GPS errors,

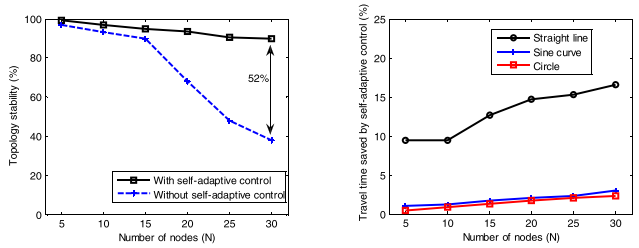


Fig. 13. Advantages of the self-adaptive control. (a) Self-adaptive control makes topology more stable when encountering obstacles. (b) Self-adaptive control saves time when there is no obstacle.

we add a random error from 0 to 5 m to each robot's position information in every iteration. Fig. 11 shows the performance of the proposed collective motion control algorithms with GPS errors, comparing to that with no GPS errors, under the without-leader scenario (similar results can be observed under the with-leader scenario). It validates the resistance of the proposed algorithm to GPS errors in that GPS errors degrade the performance very little in all the three types of paths: straight line, curve, and circle. From the experimental results, we can draw the following conclusions.

1) The GPS errors have a little impact on performance of the proposed control algorithms. The performance drop is no more than 1% under the without-leader scenario.
2) The impact of GPS errors hardly grows with the robot number. This is because the first each individual's random error is counteracted by each other in the network, and the second, the self-adaptive process further alleviates the effect of GPS errors.

Another issue we should consider is robot failure due to depletion of energy [4] and software/hardware faults. When a robot fails, it cannot communicate with its neighbors nor move any further. In Fig. 12, we show dynamics of the collective motion with a failure robot (denoted by red circle) when there is no leader. The results show that the proposed algorithms are resistant to robot failures: the swarm robots can self-heal and move along the preplanned path without the failure robot. Note that the proposed algorithms cannot guarantee resistance to robot failure. This is because followers are not aware of the preplanned path and they are guided by their neighbors with smaller hierarchical indices. For a follower, if all its neighbors with smaller hierarchical indices fail, the follower loses its direction to the destination.

5) Self-Adaptive Process: Besides alleviating the effects of GPS errors and robot failures, the self-adaptive process has two more benefits. On the one hand, when the environment is changing, especially when there appear obstacles along the predefined path, the self-adaptive process more tends to increase α_A/α_R, making the network more stable and less possible for link break. On the other hand, when there is no obstacle, the self-adaptive process tends to increase β_G/β_T, making the network move faster and thus save travel time.

We compare the topology stabilities of the moving network in the whole process of obstacle avoidance with two configurations: adopting the self-adaptive process or not. The preplanned path is straight line, and the size of the obstacle is 600 m. For the configuration without self-adaptive process, $\alpha_A/\alpha_R = 1$ and $\beta_G/\beta_T = 1$. Fig. 13(a) shows the comparison results. It shows that the self-adaptive process can significantly increase the topology stability. This increase is more notable with more robots in the network. In particular, the stability can be increased by 52% when there are 30 robots.

We also compare the travel time when there is no obstacle on the preplanned path. Fig. 13(b) shows the time saved by the self-adaptive process. It clearly shows that the self-adaptive process can save travel time, and the saved time increase with the number of robots. Enabling the self-adaptive process when the path is straight line, comparing to the other two kinds of path, saves the travel time most (up to 15%). This is because when there is neither obstacle nor turn along the path, the self-adaptive process will have more chance to increase β_G/β_T, and therefore increase the average moving speed of the network toward the destination.

V. Conclusion

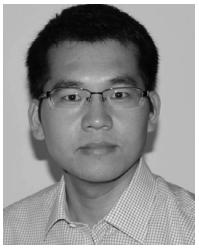
In this paper, we have studied collective motion problem in which a group of robots are requested to move along a preplanned path from a source to a destination. We have proposed self-adaptive collective motion algorithms for swarm robots in the following three cases: 1) no obstacles or leaders; 2) no obstacles with a leader; and 3) with obstacles (with and without a leader). Simulation results show the following.

1) Our algorithms enable robots to maintain connectivity and the desired neighboring distance.
2) Our algorithms are resistant to GPS errors and robot failures.
3) Self-adaptive control of our algorithms makes network topologies more stable and significantly saves the travel time.

References

[1] H. Sedjelmaci, S. M. Senouci, and N. Ansari, "A hierarchical detection and response system to enhance security against lethal cyber-attacks in UAV networks," *IEEE Trans. Syst., Man, Cybern. Syst.*, to be published.
[2] H. Çelikkanat and E. Şahin, "Steering self-organized robot flocks through externally guided individuals," *Neural Comput. Appl.*, vol. 19, no. 6, pp. 849–865, 2010.
[3] C. Cooper, D. Franklin, M. Ros, F. Safaei, and M. Abolhasan, "A comparative survey of VANET clustering techniques," *IEEE Commun. Surveys Tuts.*, vol. 19, no. 1, pp. 657–681, 1st Quart., 2017.
[4] J. Zhang *et al.*, "Energy-latency trade-off for energy-aware offloading in mobile edge computing networks," *IEEE Internet Things J.*, to be published, doi: 10.1109/JIOT.2017.2786343.
[5] I. D. Couzin, J. Krause, N. R. Franks, and S. A. Levin, "Effective leadership and decision-making in animal groups on the move," *Nature*, vol. 433, no. 7025, pp. 513–516, 2005.
[6] F. Cucker and J.-G. Dong, "Avoiding collisions in flocks," *IEEE Trans. Autom. Control*, vol. 55, no. 5, pp. 1238–1243, May 2010.
[7] (2015). *DARPA Announces Gremlins UAS Program*. Accessed: Jan. 25, 2017. [Online]. Available: http://www.unmannedsystemstechnology.com/2015/09/darpa-announces-gremlins-uas-program/
[8] M. Delight, S. Ramakrishnan, T. Zambrano, and T. Maccready, "Developing robotic swarms for ocean surface mapping," in *Proc. IEEE Int. Conf. Robot. Autom.*, May 2016, pp. 5309–5315.
[9] W. Meng, Z. He, R. Su, P. K. Yadav, R. Teo, and L. Xie, "Decentralized multi-UAV flight autonomy for moving convoys search and track," *IEEE Trans. Control Syst. Technol.*, vol. 25, no. 4, pp. 1480–1487, Jul. 2017.
[10] H. Fang, Y. Wei, J. Chen, and B. Xin, "Flocking of second-order multiagent systems with connectivity preservation based on algebraic connectivity estimation," *IEEE Trans. Cybern.*, vol. 47, no. 4, pp. 1067–1077, Apr. 2017.

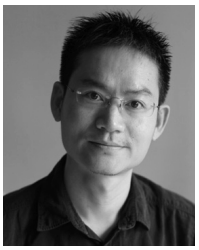
[11] E. Ferrante, A. Turgut, C. Huepe, A. Stranieri, C. Pinciroli, and M. Dorigo, "Self-organized flocking with a mobile robot swarm: A novel motion control method," *Adapt. Behav.*, vol. 20, no. 6, pp. 460–477, 2012.
[12] E. Ferrante, A. Turgut, A. Stranieri, C. Pinciroli, M. Birattari, and M. Dorigo, "A self-adaptive communication strategy for flocking in stationary and non-stationary environments," *Natural Comput.*, vol. 13, no. 2, pp. 225–245, 2014.
[13] Z. Ning *et al.*, "A cooperative quality-aware service access system for social Internet of vehicles," *IEEE Internet Things J.*, to be published, doi: 10.1109/JIOT.2017.2764259.
[14] T. T. Han and S. S. Ge, "Styled-velocity flocking of autonomous vehicles: A systematic design," *IEEE Trans. Autom. Control*, vol. 60, no. 8, pp. 2015–2030, Aug. 2015.
[15] G. Zhang, B. Shang, Y. Chen, and H. Moyes, "SmartCaveDrone: 3D cave mapping using UAVs as robotic co-archaeologists," in *Proc. Int. Conf. Unmanned Aircraft Syst. (ICUAS)*, Miami, FL, USA, Jun. 2017, pp. 1052–1057.
[16] V. J. Lumelsky and A. A. Stepanov, "Path-planning strategies for a point mobile automaton moving amidst unknown obstacles of arbitrary shape," *Algorithmica*, vol. 2, nos. 1–4, pp. 403–430, 1987.
[17] H. G. de Marina, B. Jayawardhana, and M. Cao, "Distributed rotational and translational maneuvering of rigid formations and their applications," *IEEE Trans. Robot.*, vol. 32, no. 3, pp. 684–697, Jun. 2016.
[18] M. Nagy, Z. Akos, D. Biro, and T. Vicsek, "Hierarchical group dynamics in pigeon flocks," *Nature*, vol. 464, no. 7290, pp. 890–893, 2010.
[19] R. Olfati-Saber and P. Jalalkamali, "Coupled distributed estimation and control for mobile sensor networks," *IEEE Trans. Autom. Control*, vol. 57, no. 10, pp. 2609–2614, Oct. 2012.
[20] C. W. Reynolds, "Flocks, herds and schools: A distributed behavioral model," *ACM SIGGRAPH Comput. Graph.*, vol. 21, no. 4, pp. 25–34, Jul. 1987.
[21] P. Romanczuk, I. D. Couzin, and L. Schimansky-Geier, "Collective motion due to individual escape and pursuit response," *Phys. Rev. Lett.*, vol. 102, no. 1, p. 010602, 2009.
[22] M. Rubenstein, A. Cornejo, and R. Nagpal, "Programmable self-assembly in a thousand-robot swarm," *Science*, vol. 345, no. 6198, pp. 795–799, 2014.
[23] L. Sabattini, N. Chopra, and C. Secchi, "Decentralized connectivity maintenance for cooperative control of mobile robotic systems," *Int. J. Robot. Res.*, vol. 32, no. 12, pp. 1411–1423, Sep. 2013.
[24] S. Semnani and O. A. Basir, "Semi-flocking algorithm for motion control of mobile sensors in large-scale surveillance systems," *IEEE Trans. Cybern.*, vol. 45, no. 1, pp. 129–137, Jan. 2015.
[25] K. Szwaykowska, L. Mier-y-Teran Rome, and I. B. Schwartz, "Collective motions of heterogeneous swarms," *IEEE Trans. Autom. Sci. Eng.*, vol. 12, no. 3, pp. 810–818, Jul. 2015.
[26] A. E. Turgut, H. Çelikkana, F. Gökçe, and E. Şahin, "Self-organized flocking in mobile robot swarms," *Swarm Intell.*, vol. 2, nos. 2–4, pp. 97–120, 2008.
[27] H. Liu, X. Chu, Y.-W. Leung, and R. Du, "Simple movement control algorithm for bi-connectivity in robotic sensor networks," *IEEE J. Sel. Areas Commun.*, vol. 28, no. 7, pp. 994–1005, Sep. 2010.
[28] G. Vásárhelyi *et al.*, "Outdoor flocking and formation flight with autonomous aerial robots," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, Sep. 2014, pp. 3866–3873.
[29] T. Vicsek, A. Czirók, E. Ben-Jacob, I. Cohen, and O. Shochet, "Novel type of phase transition in a system of self-driven particles," *Phys. Rev. Lett.*, vol. 75, no. 6, pp. 1226–1229, 1995.
[30] T. Vicsek and A. Zafeiris, "Collective motion," *Phys. Rep.*, vol. 517, no. 3, pp. 71–140, 2012.
[31] C. Virágh *et al.*, "Flocking algorithm for autonomous flying robots," *Bioinspiration Biomimetics*, vol. 9, no. 2, p. 025012, 2014.
[32] J. Lyu, Y. Zeng, R. Zhang, and T. J. Lim, "Placement optimization of UAV-mounted mobile base stations," *IEEE Commun. Lett.*, vol. 21, no. 3, pp. 604–607, Mar. 2017.
[33] P. Yao, Z. Xie, and P. Ren, "Optimal UAV route planning for coverage search of stationary target in river," *IEEE Trans. Control Syst. Technol.*, to be published, doi: 10.1109/TCST.2017.2781655.
[34] H.-T. Zhang, Z. Chen, and M. C. Fan, "Collaborative control of multivehicle systems in diverse motion patterns," *IEEE Trans. Control Syst. Technol.*, vol. 24, no. 4, pp. 1488–1494, Jul. 2016.
[35] M. Ke *et al.*, "Distributed TOA-based positioning in wireless sensor networks: A potential game approach," *IEEE Commun. Lett.*, vol. 22, no. 2, pp. 316–319, Oct. 2018.
[36] F. Liu, P. Shu, and J. C. S. Lui, "AppATP: An energy conserving adaptive mobile-cloud transmission protocol," *IEEE Trans. Comput.*, vol. 64, no. 11, pp. 3051–3063, Nov. 2015.
[37] F. Liu *et al.*, "Gearing resource-poor mobile devices with powerful clouds: Architectures, challenges, and applications," *IEEE Wireless Commun.*, vol. 20, no. 3, pp. 14–22, Jun. 2013.
[38] Y. Jin, F. Liu, X. Yi, and M. Chen, "Reducing cellular signaling traffic for heartbeat messages via energy-efficient D2D forwarding," in *Proc. IEEE ICDCS*, Atlanta, GA, USA, Jun. 2017, pp. 1301–1311.
[39] T. Zhang, X. Zhang, F. Liu, H. Leng, Q. Yu, and G. Liang, "eTrain: Making wasted energy useful by utilizing heartbeats for mobile data transmissions," in *Proc. IEEE ICDCS*, Columbus, OH, USA, Jun. 2015, pp. 113–122.
[40] Simulator. Accessed: Jun. 15, 2018. [Online]. Available: http://www.comp.hkbu.edu.hk/~chxw/CollectiveMotion.zip



Haitao Zhao (SM'18) received the B.E., M.Sc., and Ph.D. degrees from the National University of Defense Technology (NUDT), Changsha, China.

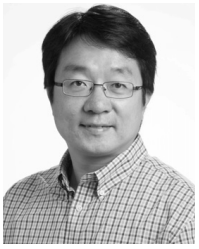
He visited the Institute of ECIT, Queen's University of Belfast, Belfast, U.K., and Hong Kong Baptist University, Hong Kong. He is currently an Associate Professor with the Department of Cognitive Communications, NUDT. His current research interests include cognitive radio networks and self-organized networks.

Prof. Zhao served as a TPC member for the IEEE ICC'14-17 and Globecom'16-17, and the Guest Editor of the IEEE COMMUNICATIONS MAGAZINE.



Hai Liu (M'16) received the B.Sc. and M.Sc. degrees in applied mathematics from the South China University of Technology, Guangzhou, China, and the Ph.D. degree in computer science from the City University of Hong Kong, Hong Kong.

He held several academic posts at the University of Ottawa, Ottawa, ON, Canada, and Hong Kong Baptist University, Hong Kong. He is currently an Assistant Professor with the Department of Computing, Hang Seng Management College, Hong Kong. His current research interests include wireless networking, cloud computing, and algorithm design and analysis. His H-index is 22 according to Google Scholar.



Yiu-Wing Leung received the B.Sc. and Ph.D. degrees from the Chinese University of Hong Kong, Hong Kong.

He has been with Hong Kong Baptist University, Hong Kong, where he is currently a Professor with the Computer Science Department and the Programme Director of two M.Sc. programs. He has authored over 80 papers in various IEEE journals and conferences. His current research interests include three major areas: 1) network design, analysis, and optimization; 2) Internet and cloud computing; and 3) systems engineering and optimization.

Xiaowen Chu (SM'11) received the B.E. degree in computer science from Tsinghua University, Beijing, China, in 1999, and the Ph.D. degree in computer science from the Hong Kong University of Science and Technology, Hong Kong, in 2003.

He is currently an Associate Professor with the Department of Computer Science, Hong Kong Baptist University, Hong Kong. His current research interests include distributed and parallel computing and wireless networks.

Dr. Chu is currently an Associate Editor of the IEEE ACCESS and the IEEE INTERNET OF THINGS JOURNAL.