

Autonomous Navigation of Wireless Robot Swarms with Covert Leaders

Xiaofeng Han[◊] Louis F. Rossi^{*} Chien-Chung Shen[◊]

Abstract—The integration of advanced computation, wireless communication, and control technologies has facilitated the creation of autonomous robot swarms for many civil and military applications. In nature, animals that travel in groups often rely on social interactions among group members to make movement decisions. In many cases, few individuals within the group have pertinent knowledge about the destination and/or migration routes. In this paper, we adapt a swarm model developed for animal groups to study the unique problems associated with *covert leadership* in the context of wireless robot swarms. We term this problem *autonomous navigation with covert leaders*. In this covert leadership problem, only a small subset of robots in a robot swarm possess extra information that guides their movement, and both this information and the identities of those individuals possessing this information remain covert (to minimize the chance of being compromised). We describe a distributed navigation algorithm, where each robot locally makes its movement decision solely based on one-hop information collected via wireless communications. The effectiveness and merits of the described navigation algorithm are demonstrated through extensive simulations.

I. INTRODUCTION

Animals that travel in groups often rely on social interactions among group members to make movement decisions. In many cases, few individuals within the group have pertinent knowledge about the destination and/or migration routes. Based on such observation, Couzin et al. [1] has developed a swarm model showing how information can be transferred within animal groups both without explicit signaling and when group members do not know which individuals possess such pertinent knowledge.

The integration of advanced computation, wireless communication, and control technologies has facilitated the creation of autonomous robot swarms for many civil and military applications, where groups of robots executing swarm algorithms self-organize to achieve different goals. A swarm algorithm is loosely defined as a set of rules which a group of robots follow to interact locally with other proximal robots without any centralized control. In particular, certain sets of rules in certain swarm algorithms self-organize the group into coherent, useful structures and behavior such as a uniformly translating

mass, traveling in a fixed direction, or a milling mass which holds a fixed position.

In this paper, we adapt the swarm model described in [1] to study the unique problems associated with *covert leadership* in the context of wireless robot swarms. We term this problem *autonomous navigation with covert leaders*. Covert leadership in a wireless robot swarm involves long-standing problems associated with swarm algorithms and introduces new problems and challenges unique to wireless communications that do not exist in biological contexts. In typical leadership problems in swarms, a small subset of individuals possess extra information that guides their movement. For security and other concerns, both this information and the identities of those individuals possessing this information should be kept covert. Therefore, it is critical that this information not be broadcast via wireless communications and it is best to have the covert information on as few robots as possible (to minimize the chance of being compromised). The key differences (between biological swarms and robot swarms) are wireless definitions of robot proximity, decision making data arriving at discrete intervals, and covert leadership as a desirable quality. All swarm algorithms rely upon the notion of proximal interactions, but proximity in most biological systems is defined through direct sensory input *e.g.*, visual contact. In contrast, proximity associated with wireless communications is determined by transmission power, channel fading, path loss, among other factors. The second difference is that decision making data in biological systems arrive continuously in time or essentially continuously since nervous system respond on a time scale faster than the motion of the individual in most cases, whereas data arrives in discrete bursts in a wireless system. The last difference is that we consider a unique scenario where the information possessed by leaders is absolutely covert in contrast to biological systems where it is possible and in some cases likely that leaders can be identified through sensory or behavioral cues.

Swarm techniques have been developed to control robots, such as pattern formation [2], [3] and navigation [4]. However, these swarm solutions either do not utilize wireless communications as the coordination methods or assumes that robots can continuously exchange information without any communication collision or packet loss. Furthermore, most existing work on autonomous

[◊]Department of Computer and Information Sciences / ^{*}Department of Mathematical Sciences, University of Delaware, U.S.A. {han,cshen}@cis.udel.edu,rossi@math.udel.edu.

navigation focuses on computing trajectories satisfying certain constraints, such as collision-free or obstacle avoidance, for each individual robot. For instance, the trajectories can be computed globally assuming that the working environment is completely known [5], or computed locally by each robot using information collected in real-time in dynamic environment [6], [7]. However, these efforts assume that the pertinent information related to routes and destinations is known to all the robots. In addition, these efforts do not use wireless communications as the coordination method among robots.

Our solution is very different from existing work on autonomous navigation in that we apply covert leadership to control the movement of wireless robot swarms. Our algorithm aims to navigate a robot swarm from one location to another while satisfying the follow requirements: (1) the number of leaders should be as small as possible, (2) the leaders' identities should remain covert, and (3) the destination information should remain secret. The proposed navigation algorithm is fully distributed, and each robot executes the algorithm asynchronously using information collected via wireless communications from one-hop neighbors.

The remainder of the paper is organized as follows. Section II briefly reviews the navigation model presented [1] for animal groups. Section III presents our navigation algorithm for wireless robot swarms. Performance of the presented algorithm is evaluated via extensive simulations in Section IV. Section V concludes the paper with future research directions.

II. BASIC SWARM MODEL

This section briefly reviews the navigation model for animal groups as described in [1]. The perceptual field of each animal is divided into zone of repulsion (ZOR), zone of orientation (ZOO) and zone of attraction (ZOA), as shown in Fig. 1. Given a distribution of N animals, to coordinate with the neighbors in different zones, animal i (located at position vector P_i and pointing in direction D_i) will move *away* from the neighbors in ZOR or move *along* with the neighbor in ZOO while moving *towards* the neighbors in ZOA, and therefore will have three decision vectors,

$$DR_i = \sum_{j \in S_{ZOR}} \frac{R_{ij}}{|R_{ij}|}, \quad DO_i = \sum_{j \in S_{ZOO}} \frac{D_j}{|D_j|}, \quad DA_i = \sum_{j \in S_{ZOA}} \frac{R_{ji}}{|R_{ji}|} \quad (1)$$

where $R_{ij} = P_i - P_j$ is a displacement vector between the i^{th} and j^{th} animal, and S_{ZOR} , S_{ZOO} and S_{ZOA} are the sets of indices of animals in the zones of repulsion, orientation and attraction, respectively. Similarly, $|S_Z|$ denotes the number of elements in zone Z . The normalization of the decision vectors is not discussed in [1], but we assume that each decision is treated as a unit vector in the network simulations.

If we assume that decision vectors are normalized as unit vectors then the biologically inspired swarming algorithm introduced in [1] is as follows.

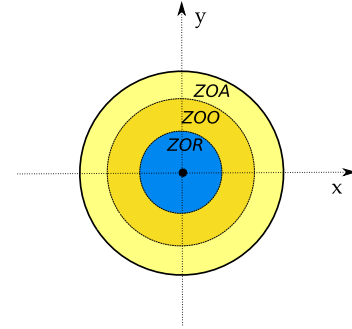


Fig. 1. Representation of single animal in the navigation model. The perceptual field is divided into ZOR, ZOO and ZOA from inside to outside.

- 1) If $|S_{ZOR}| \neq 0$ then $V_i = DR_i$. Break.
- 2) If $|S_{ZOO}| \neq 0$ and $|S_{ZOA}| = 0$ then $V_i = DO_i/|DO_i|$. Break.
- 3) If $|S_{ZOO}| = 0$ and $|S_{ZOA}| \neq 0$ then $V_i = DA_i/|DA_i|$. Break.
- 4) If $|S_{ZOO}| \neq 0$ and $|S_{ZOA}| \neq 0$ then $V_i = \alpha \times DO_i/|DO_i| + (1 - \alpha) \times DA_i/|DA_i|$.

It is important to note that other normalizations are possible and some are more effective than others in producing self-organized, coherent behavior. Typically, α is chosen to be $1/2$. Enhancements of the basic model include limited cones of perception and limited turning rates. Another interesting feature of this basic swarming algorithm is that the decision vectors are spatially discontinuous. This means that slight changes in relative positions may cause the direction vector to change abruptly responding to animals crossing zone boundaries. This is not a major issue in biological applications where the size of the zone of repulsion is relatively small. In mobile networking applications one desires well-spaced network to achieve maximal coverage. In fact, a desired zone of repulsion can be as much as 30% of the transmission radius. In simple discontinuous models like these, high swarm densities lead to unstable behavior and no self-organization will occur.

In leadership studies, there are two types of animals, leaders and non-leaders. Leaders possess extra information about the environment. In our study, the leaders possess a vector $F_i = P_d - P_i$ pointing to the desired destination, where P_d is the position vector of destination. Leaders will set their new orientation to be

$$D_i = \beta \times V_i + (1 - \beta) \times F_i. \quad (2)$$

where typically β is chosen to be $1/2$. Non-leaders will set their new orientation to be

$$D_i = V_i. \quad (3)$$

While only a small fraction of the animals possess destination information, the swarming rules effectively communicate this information to non-leaders through the rule-based dynamics of the swarm. Under proper

conditions, the leaders will guide non-leader neighbors toward a consensus direction through covert leadership. Note that each animal, either leader or non-leader, will keep observing its proximity and continuously use the coordination rules to interact with the neighbors within different zones. Simulation results in [1] indicate that a small percentage leaders can effectively lead the group to the destination point.

III. DISTRIBUTED NAVIGATION ALGORITHM FOR WIRELESS ROBOT SWARMS

Although the navigation model described in [1] effectively mimics the covert leadership behavior existing in animal groups, this model cannot be directly applied to wireless robot swarms due to the constraints imposed by wireless communications. We therefore present a distributed and asynchronous autonomous navigation algorithm designed specifically for wireless robot swarms.

A. System Model of Wireless Robot Swarms

The system model of wireless robot swarms makes the following assumptions.

Assumption 1: A wireless robot swarm is composed of ‘homogeneous’ robots, where each robot executes the same swarm algorithm and possesses identical omnidirectional wireless communication capability.

Assumption 2: Since the one-hop propagation delay in wireless communications is usually very low (in the order of milliseconds), the propagation delay is ignored such that each broadcast packet can be received immediately after its transmission.

Assumption 3: Each robot is aware of its own position at all time. Such position information can be obtained via GPS or other localization algorithms. The position of the destination is known only to a small set of covert leaders within the robot swarm.

Assumption 4: Each robot moves at the same speed V and can turn its moving direction to any direction it desires. However, the model is readily to be extended to the situations where robots move at different speed or have limited maximum turning angle.

B. Distributed Autonomous Navigation Algorithm

Adopting the idea of [1], the transmission area of each robot (viewed as a circle centered at the robot with radius being equal to the transmission radius) is divided into zone of repulsion, zone of orientation, and zone of attraction, respectively. However, to facilitate interactions between ‘neighboring’ robots using these three zones, we need to address the following two critical issues. First, characteristics of wireless communications make it impossible for a robot to continuously receive the position information from all of its neighbors. Robots can only exchange information periodically. Second, potential collisions of wireless communications make it

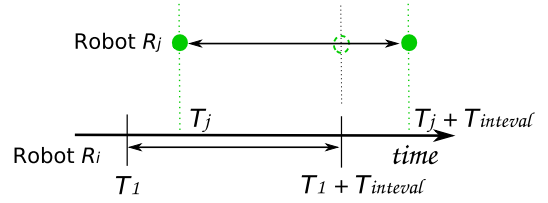


Fig. 2. Example of estimating the positions of neighbors.

impossible for a robot to receive the position information from all of its neighbors at the exact same time. As a result, when a robot needs to decide its moving direction according to the coordination rules, it must be able to estimate the current positions and moving directions of its neighbors based on previously received information.

To address these issues, each robot *independently* divides the time into time slots of fixed length. At the beginning of a new time slot, based on the information collected in the previous time slot, each robot calculates its moving direction according to the coordination rules and immediately broadcasts a hello packet containing its current position and new moving direction. Then each robot will move along the decided direction until the next time slot. How the time slots help each robot to estimate the updated information about its neighbors is depicted in Figure 2. Each robot, say R_i , maintains a neighbor cache, which is reset to empty at the beginning of each time slot. During a particular time slot starting at time T_1 , upon R_i 's receiving a hello packet from robot R_j at local time T_j , R_i records the position P_{j1} and the moving direction D_j of R_j together with timestamp T_j in its neighbor cache. Using the information in the neighbor cache, R_i can estimate the new position P_{j2} of R_j at the beginning of R_i 's next time slot, *i.e.*, time $T_1 + T_{interval}$, using Equation 4, since R_j is assumed to move along D_j until $T_j + T_{interval}$, which is later than $T_1 + T_{interval}$.

$$P_{j2} = P_{j1} + (T_1 + T_{interval} - T_j) \times \frac{D_j}{|D_j|} \times V \quad (4)$$

Algorithm 1 presents the autonomous navigation algorithm, and the notations used in the algorithm are listed in Table I. Notice that Algorithm 1 is an event-driven algorithm, and each robot executes this algorithm independently until the entire swarm arrives at the destination. Moreover, since the algorithm only employs the local time of each robot, there is no need to synchronize the entire swarm.

IV. SIMULATION STUDY

We conduct extensive simulation to evaluate the performance of the proposed autonomous navigation algorithm with respect to two evaluation metrics, average arrival ratio and average arrival time, whose formal definitions are given below. In the simulation, the initial robot swarms are randomly deployed with a density of 50 robots per square kilometer. We use QualNet

Algorithm 1 Autonomous Navigation Algorithm

1: **OUTPUT:** Robot R_i periodically computes its moving direction D_i ;

2: **START UP**

3: R_i waits for a random time T_{i0} between $[0, T_{jitter}]$.

4: **if** (R_i is a leader robot) **then**

5: Set D_i to be the direction $\frac{P_d - P_i}{|P_d - P_i|}$ towards the destination;

6: **else**

7: Set D_i to be a random direction;

8: **end if**

9: R_i broadcasts a hello packet containing its position P_i and moving direction D_i ;

10: R_i resets its neighbor cache, schedules a timer for next time slot starting at $T_{i1} = T_{i0} + T_{interval}$, and starts moving along D_i ;

11: **END START UP**

12: **UPON EVENT** *Receive hello packet from R_j at T_{ij} :*

13: R_i records the position P_j and moving direction D_j of R_j together with time stamp T_{ij} in its neighbor cache;

14: **END UPON EVENT**

15: **UPON EVENT** *Timer ticks for new time slot at T_{in} :*

16: $D_i = DR_i = DO_i = DA_i = null$;

17: **for** each record R_j in the neighbor cache **do**

18: R_i computes the distance $|R_i R_j|$;

19: **if** ($|R_i R_j| \leq R_{zor}$) **then**

20: $DR_i = DR_i + \frac{P_i - P_j}{|P_i - P_j|}$;

21: **end if**

22: **if** ($R_{zor} < |R_i R_j| \leq R_{zoo}$) **then**

23: $DO_i = DO_i + \frac{D_j}{|D_j|}$

24: **end if**

25: **if** ($R_{zoo} < |R_i R_j| \leq R_{zoa}$) **then**

26: $DA_i = DA_i + \frac{P_j - P_i}{|P_j - P_i|}$

27: **end if**

28: **end for**

29: **if** ($NR \neq null$) **then**

30: $D_i = \frac{VR}{|VR|}$;

31: **else**

32: $D_i = \alpha \times \frac{RO_i}{|RO_i|} + (1 - \alpha) \times \frac{RA_i}{|RA_i|}$;

33: **if** (R_i is a leader robot) **then**

34: $D_i = \beta \times \frac{D_i}{|D_i|} + (1 - \beta) \times \frac{P_d - P_i}{|P_d - P_i|}$;

35: **end if**

36: **end if**

37: R_i broadcasts a hello packet containing its position P_i and moving direction D_i ;

38: R_i resets neighbor cache, schedules a timer for next time slot starting at $T_{in+1} = T_{in} + T_{interval}$, and starts moving along D_i ;

39: **END UPON EVENT**

T_{jitter}	Initial jitter
$T_{interval}$	Length of each time slot
R_{zor}	Radius of ZOR
R_{zoo}	Radius of ZOO
R_{zoa}	Radius of ZOA
$ R_i R_j $	The distance between robot R_i and robot R_j
P_i	The position vector of robot R_i
P_d	The position vector of the destination
D_i	The moving direction vector of R_i
DR_i, DO_i, DA_i	Vectors of R_i for its neighbors in ZOR, ZOO and ZOA
α	Constant within $[0, 1]$
β	Constant within $[0, 1]$

TABLE I
TERMS AND THEIR SEMANTICS IN ALGORITHM 1

PHYSICAL and MAC Layer Protocols	802.11b
Data Rate (MBPS)	5
Transmission Range (Meters)	354
Broadcast Interval (Seconds)	2
Initial Jitter (Seconds)	1
Number of Robots	10 - 50
Speed (MPS: Meters / second)	1 - 10
Percentage of Leaders (%)	10 - 25
α	0.5
β	0.5

TABLE II
SIMULATION CONFIGURATIONS

as the simulation platform. Table II summarizes the configurations of these simulations. All of the qualitative results presented in this section are the average of 100 trials.

Definition 4.1: [Successful Arrival]: A robot swarm successfully arrives at the destination if the swarm is connected and the destination point is within the transmission area of at least one robot leader.

Definition 4.2: [Average Arrival Ratio]: For a robot swarm using a given configuration and moving to the destination, its average arrival ratio is the number of successful arrivals over the number of total simulation runs.

Definition 4.3: [Average Arrival Time]: For a robot swarm using a given configuration and moving to the destination, the average arrival time is the average time that the swarm needs to arrive at the destination point over all of the successful arrivals.

A. Evaluation of the Size of the Three Zones

In the first set of simulations, we set the moving speed of each robot to be 5 MPS, and let a wireless robot swarm containing 30 robots with 15% leaders move to a destination point about 5 kilometers away. We repeat this simulation with different configurations of R_{ZOR} , R_{ZOO} and R_{ZOA} .

Figures 3(a), 3(b) and 3(c) demonstrate the navigation process of the robot swarm using $R_{ZOR} = 40$ meters, $R_{ZOO} = 274$ meters and $R_{ZOA} = 40$ meters. It takes about 1180 seconds for the swarm to get to the destination. Once arriving at the destination, the entire swarm will move around the destination point, as shown in 3(c).

Figures 4(a) and 4(b) compare the performance of the robot swarm using different combinations of R_{ZOR} , R_{ZOO} and R_{ZOA} (We increase R_{ZOR} and R_{ZOA} from 0% to 10% of the transmission radius, respectively). First, when the sizes of R_{ZOR} and R_{ZOA} are less than 10% of the total transmission range (area 1 in figure 4(a)), which implies a large R_{ZOO} about 80% of the transmission range, the robot swarms can quickly move to the destination with 100% average arrival ratio. Second, as shown in area 2 in figure 4(a), for R_{ZOR} ranging from 0% to 20% of the transmission range, there is a valid range of R_{ZOA} ranging from 5% to 70% of the transmission range, within which the robot swarms can arrive at the destination with arrival ratio 100%. Also, when the size of R_{ZOR} increases, the valid range of R_{ZOA} decreases correspondingly. Third, when R_{ZOR} is larger than 22.5% of the transmission radius, the robot swarms fail to arrive at the destination most of the times. However, exceptions occur when R_{ZOR} is between 50% and 80% of the transmission radius, where the robot swarms can actually arrive at the destination with high average arrival ratio by choosing a small R_{ZOA} . Such results deserve more explanation. We track the movement of the swarms and find that when R_{ZOR} is within this particular range, with small R_{ZOO} and R_{ZOA} , the robot swarms can still adjust to consensus moving directions towards the destination and hence eventually arrive at the destination. However, after successfully arriving at the destination, if no further action is taken, only leader robots may stay, while non-leader robots will continue moving and result in partition. Finally, we also notice that when the average arrival ratio decreases, the average arrival time increases correspondingly.

B. Evaluation of the Number of Robots

We then evaluate how the number of robots in the swarms would affect the performance of our navigation algorithm. In these simulations, each robot uses $R_{ZOR} = 40$ meters, $R_{ZOO} = 274$ meters and $R_{ZOA} = 40$ meters. We gradually increase the number of robots from 10 to 50, and let the swarms move to a destination about 5 kilometers away. We test the presented navigation algorithm in different configurations using leader percentage 15%, 20% and 25% coupled with moving speed 3 MPS, 6 MPS and 9 MPS. Figure 5 shows the average arrival ratio. As we can see, when the moving speed is low (3 MPS), the average arrival ratio is 100%. Such results remain stable when the number of robots increases except for the cases of large swarms containing 50 robots and 15% or 20% leaders. However, when the moving speed is high (6MPS and 9 MPS), the average arrival ratio drops significantly when the number of robots is over a threshold number (40 when the moving speed is 6 MPS; 35 when the moving speed is 9 MPS). We also notice that the increase of leader percentage can effectively improve the average arrival ratio of large

robot swarms. Figure 6 shows the average arrival time. We observe that the more robots a swarm has, the more time the swarm requires to arrive at the destination. We also notice that once the number of robots increases beyond a threshold number when moving at 6 MPS or 9 MPS, the average arrival time increases sharply. This is because when the moving speed is high, it takes much more time for large robot swarms to coordinate and make consensus decisions on the moving direction. Finally, when the moving speed is low, the different leader percentage would not improve the average arrival time; however, when the moving speed is high, more leaders could effectively reduce the average arrival time of large robot swarms.

C. Evaluation of the Percentage of Leaders

The percentage of leaders in the robot swarms is certainly an important parameter in the presented navigation algorithm. In the third set of simulations, we set the destination point to be 5 kilometers away, and let each robot use $R_{ZOR} = 40$ meters, $R_{ZOO} = 274$ meters and $R_{ZOA} = 40$ meters. Then we gradually increase the percentage of leaders from 10% to 25% to evaluate the performance of the navigation algorithm in robot swarms containing 10, 20, 30, 40, 50 robots and moving at the speed of 3 MPS, 6 MPS and 9 MPS. Figure 7 shows the average arrival ratio. For small swarms containing 30 or less robots, the arrival ratio is always 100%, and such result remain stable with different leader percentage or moving speed; For large swarms containing 40 or more robots, the average arrival ratio is low initially. However, it can be effectively improved with increase of leader percentage, especially when each robot moves at a high speed of 6 MPS or 9 MPS. Figure 8 shows the corresponding average arrival time of Figure 7. In general, the faster the robots move, the quicker robot swarms arrive at the destination. However, we observe fluctuations for the robot swarm containing 50 robots, for these particular robot swarms, their average arrival time is unstable when the moving speed is 9 MPS, since the average arrival ratio is very low.

D. Evaluation of the Moving Speed

Finally, we evaluate the presented navigation algorithm using different moving speed. In these simulations, each robot uses $R_{ZOR} = 40$ meters, $R_{ZOO} = 274$ meters and $R_{ZOA} = 40$ meters. We gradually increase the moving speed from 1 MPS to 10 MPS, and compare the performance of our algorithm in robot swarms containing 10, 20, 30, 40 and 50 robots, and having 15%, 20% and 25% leaders. Again, the destination point is about 5 kilometers away. Figure 9 shows the average arrival ratio of the robot swarms. For small swarms containing 30 or less robots, the average arrival ratios remain stable when the moving speed increases. However, for large

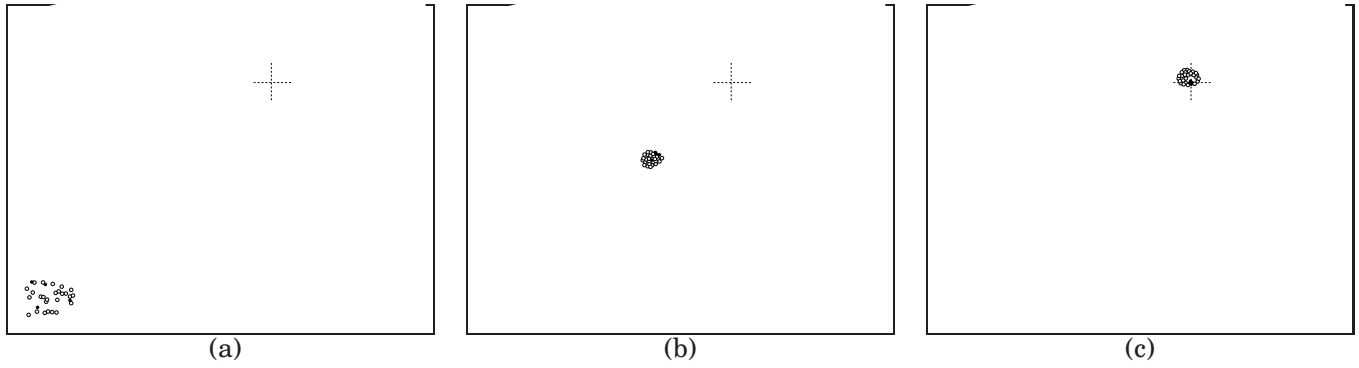


Fig. 3. The hollow circles stand for the non-leader robots, while the solid circles stand for the leader robots. The dot-line crossing in the up-right corner is the destination. (a) The initial topology of the wireless robot swarm. (b) The topology of the wireless robot swarm at 800 seconds. (c) The topology of the wireless robot swarm at 1500 seconds.

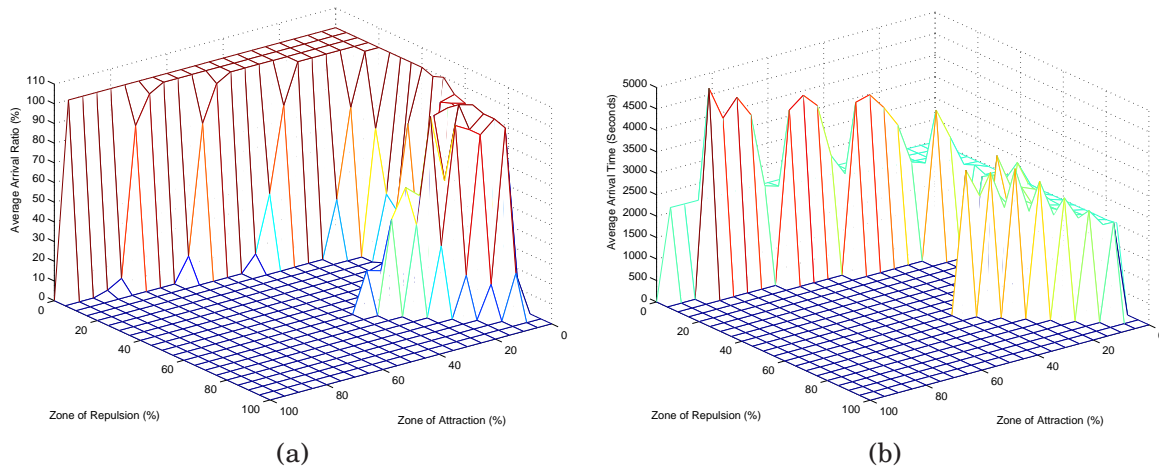


Fig. 4. Average arrival ratio and arrival time of a robot group containing 30 robots with 15% leaders moving to a destination point about 5 kilometers away. (a) Average arrival ratio. (b) Average arrival time.

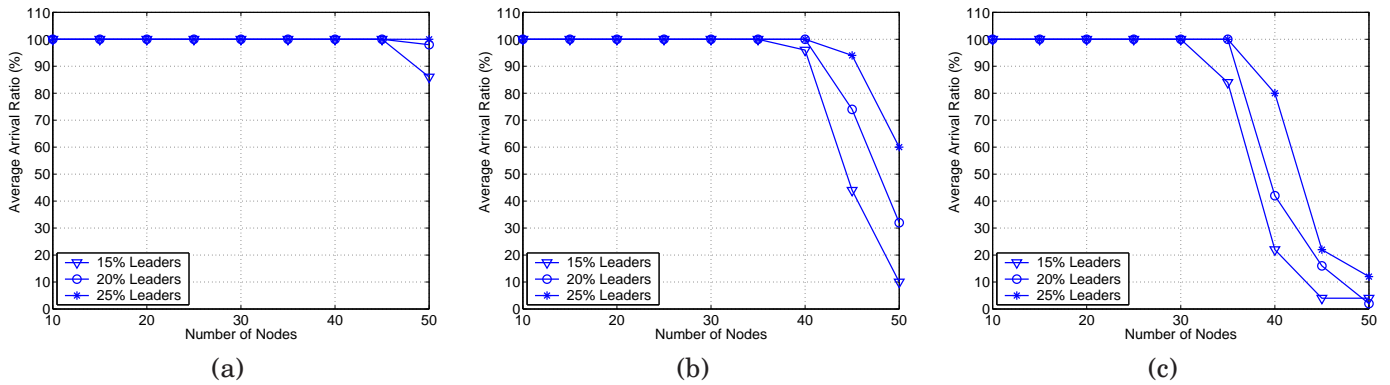


Fig. 5. Average arrival ratio of the wireless robot swarms containing 15%, 20% and 25% leaders and moving at the speed of 3 MPS, 6 MPS and 9 MPS, respectively. (a) Moving Speed = 3 MPS. (b) Moving Speed = 6 MPS. (c) Moving Speed = 9 MPS.

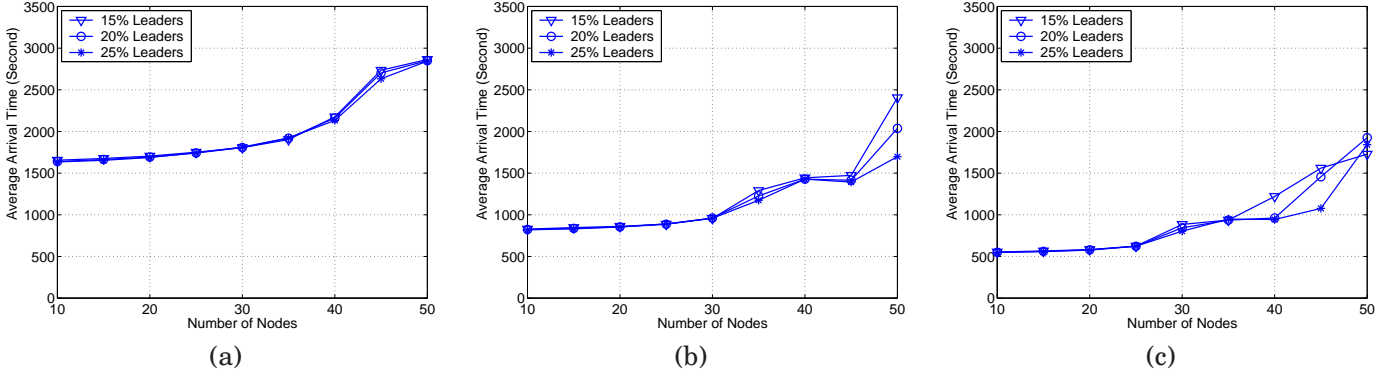


Fig. 6. Average arrival time of the wireless robot swarms containing 15%, 20% and 25% leaders and moving at the speed of 3 MPS, 6 MPS and 9 MPS, respectively. (a) Moving Speed = 3 MPS. (b) Moving Speed = 6 MPS. (c) Moving Speed = 9 MPS.

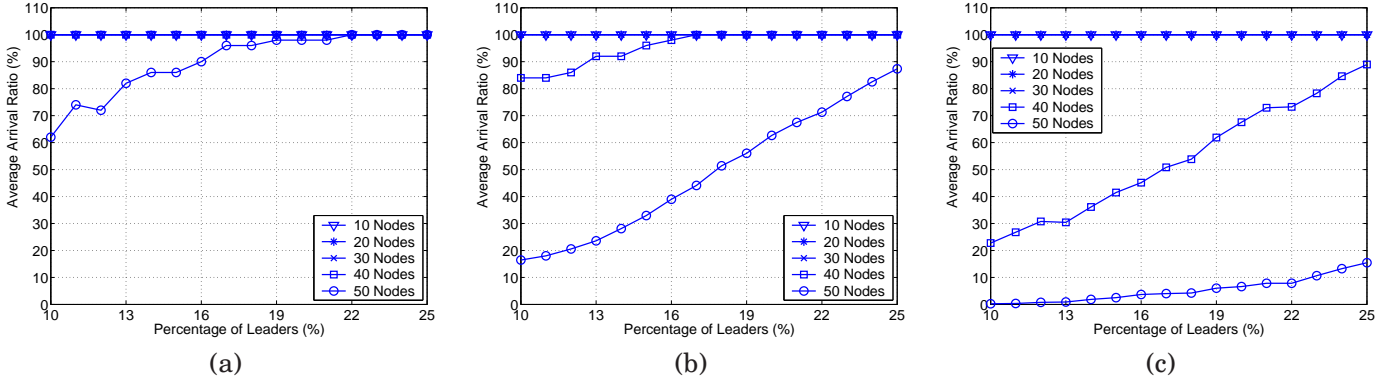


Fig. 7. Average arrival ratio of wireless robot swarms containing 10, 20, 30, 40, and 50 robots and moving at the speed of 3 MPS, 6 MPS and 9 MPS. (a) Moving Speed = 3 MPS. (b) Moving Speed = 6 MPS. (c) Moving Speed = 9 MPS.

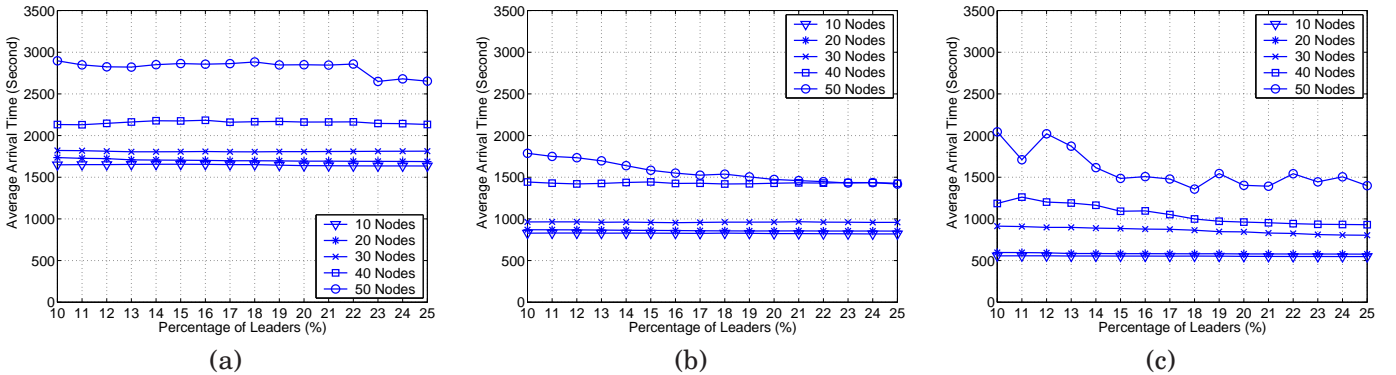


Fig. 8. Average arrival time of wireless robot swarms containing 10, 20, 30, 40, and 50 robots and moving at the speed of 3 MPS, 6 MPS and 9 MPS. (a) Moving Speed = 3 MPS. (b) Moving Speed = 6 MPS. (c) Moving Speed = 9 MPS.

swarms with 40 or more robots, the increase of moving speed can significantly degrade the average arrival ratio when the leader percentage is low. We also observe that this negative effect can be effectively improved if the leader percentage increases. Figure 10 shows the average arrival time of the robot swarms. In general, the more robots a swarm has, the more time the swarm needs to arrive at the destination. Furthermore, the increase of leader percentage would not improve the average arrival time.

E. Summary of Simulation Results

The key results of the all of the simulations can be summarized as follows. (1) When using proper configurations, the presented navigation algorithm can effectively lead the robot swarms to the destination. Such results remain stable within a wide range of swarm size, moving speed and leader percentage. (2) For large robot swarms containing 40 or more robots, the successful arrival ratio drops when the moving speed is high or the leader percentage is low. This implies that for a given robot swarm where each robot uses fixed radii of ZOR, ZOO and

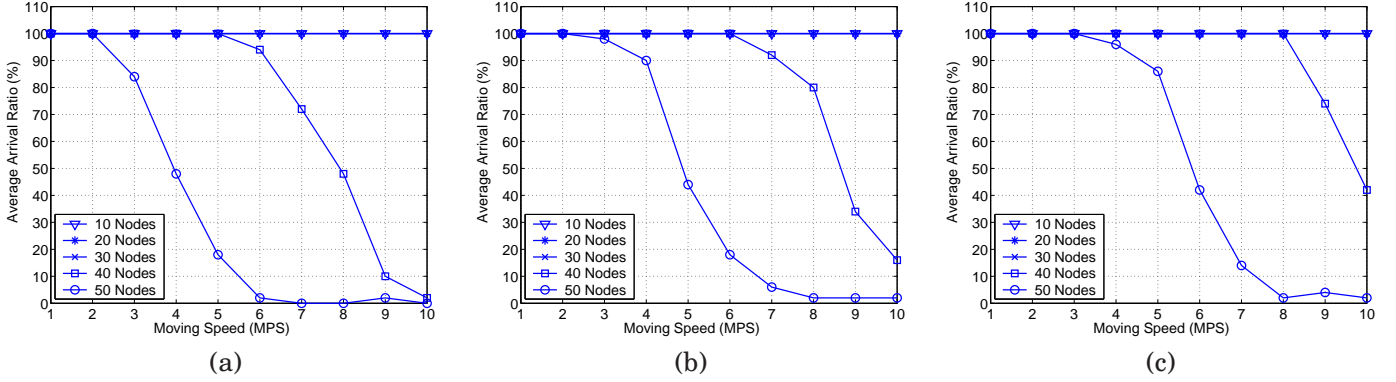


Fig. 9. Average arrival ratio of wireless robot swarms containing 10, 20, 30, 40, and 50 robots and having 15%, 20%, and 25% leaders, respectively. (a) Leader Percentage = 15%. (b) Leader Percentage = 20%. (c) Leader Percentage = 25%.

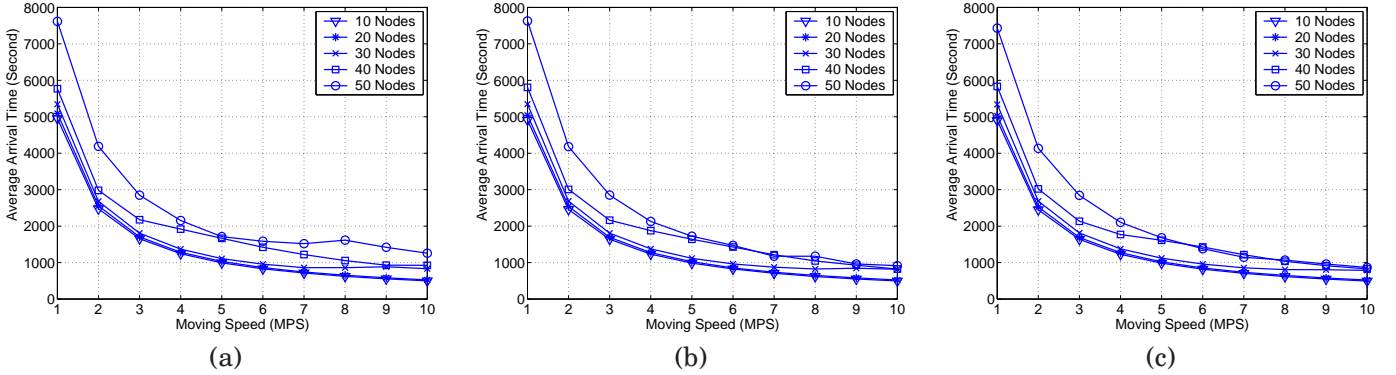


Fig. 10. Average arrival ratio of wireless robot swarms containing 10, 20, 30, 40, and 50 robots and having 15%, 20%, and 25% leaders, respectively. (a) Leader Percentage = 15%. (b) Leader Percentage = 20%. (c) Leader Percentage = 25%.

ZOA, there is a threshold moving speed and a threshold leader percentage in the presented algorithm. In order to achieve better performance, the moving speed must be lower than the threshold value or the leader percentage must be higher than the threshold value. This points out an important future research issue: to derive the thresholds of moving speed and leader percentage using mathematical analysis.

V. CONCLUSION AND FUTURE WORK

This paper presents an autonomous navigation algorithm for wireless robot swarms. Unlike traditional navigation problems and solutions, our model aims to navigate robot swarms using covert leaders where only a small number of leader robots possess the destination information, and both the leaders' identities and the destination information should remain secret at all time. Our model is fully distributed and robots coordinate via wireless communications without any synchronization. Simulation results show that through local coordinations, covert leaders can effectively help robot swarms make a consensus decision on moving direction and thus lead the robot swarms to successfully arrive at the destination. Future research includes mathematically analyzing the thresholds of moving speed and leader percentage for the given configuration of robot swarms,

as well as extending our algorithm to more complicated scenarios, such as each robot has kinetic constraints and static or mobile obstacles exist in the environment.

REFERENCES

- [1] I. Couzin, J. Krause, N. Franks, and S. Levin, "Effective leadership and decision-making in animal groups on the move," *NATURE*, vol. 433 (7025), pp. 513–516, 2005.
- [2] N. Michael, C. Belta, and V. Kumar, "Controlling three dimensional swarms of robots," in *Proceedings of IEEE International Conference on Robotics and Automation (ICRA)*, 2006.
- [3] D. V. Dimarogonas and K. J. Kyriakopoulos, "A connection between formation control and flocking behavior in nonholonomic multi-agent systems," in *Proceedings of IEEE International Conference on Robotics and Automation (ICRA)*, 2006.
- [4] S. Berman, A. Halasz, V. Kumar, and S. Pratt, "Bio-inspired group behaviors for the deployment of a swarm of robots to multiple destinations," in *Proceedings of IEEE International Conference on Robotics and Automation (ICRA)*, 2007.
- [5] T.-Y. Li and H.-C. Chou, "Motion planning for a crowd of robots," in *Proceedings of IEEE International Conference on Robotics and Automation (ICRA)*, vol. 3, 2003, pp. 4215–4221.
- [6] K. Macek, M. Becker, and R. Siegwart, "Motion planning for car-like vehicles in dynamic urban scenarios," in *Proceedings of IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2006, pp. 4375–4380.
- [7] V. John and X. Jing, "Real-time motion planning of multiple mobile manipulators with a common task objective in shared work environments," in *Proceedings of IEEE International Conference on Robotics and Automation (ICRA)*, 2007.