18

Flocking Control for Swarms of Mobile Robots Inspired by Fish Schools

Geunho Lee and Nak Young Chong
School of Information Science, Japan Advanced Institute of Science and Technology
Japan

1. Introduction

Self-organizing and adaptive behaviors can be easily seen in flocks of birds or schools of fish. It is surprising that each individual member follows a small number of simple behavioral rules, resulting in sophisticated group behaviors (Wilson, 2000). For instance, when a school of fish is faced with an obstacle, they can avoid collision by being split into a plurality of smaller groups that can be merged after they pass around the obstacle. Based on the observation of such habits of schooling fishes, we propose collective navigation behavior rules that enable a large swarm of autonomous mobile robots to flock toward a stationary or moving goal in an unknown environment. Recently, robot swarms are expected to be deployed in a wide variety of applications such as odor localization, mobile sensor networking, medical operations, surveillance, and search-and-rescue (Sahin, 2005). In order to perform those tasks successfully, the behaviors of individual robots need to be controlled in a simple manner to support coordinated group behavior.

Reynolds presented a distributed behavioral model of coordinated animal motion based on fish schools and bird flocks (Reynolds, 1987). His work demonstrated that navigation is an example of emergent behavior arising from simple rules. Many navigation strategies reported in the field of swarm robotics can be classified into centralized and decentralized strategies. Centralized strategies (Egerstedt & Hu, 2001) (Burgard *et al*, 2005) employ a central unit that organizes the behaviors of the whole swarm. This strategy usually lacks scalability and becomes technically unfeasible when a large swarm is considered. On the other hand, decentralized strategies are based on interactions between individual robots mostly inspired by evidence from biological systems or natural phenomena. Decentralized strategies can be further divided into biological emergence (Baldassarre *et al*, 2007) (Shimizu *et al*, 2006) (Folino & Spezzano, 2002), behavior-based (Ogren & Leonard, 2005) (Balch & Hybinette, 2000), and virtual physics-based (Spears et al, 2006) (Esposito & Dunbar, 2006) (Zarzhitsky et al, 2005) approaches. Specifically, the behavior-based and virtual physics-based approaches are related to the use of such physical phenomena as crystallization (Balch & Hybinette, 2000) gravitational forces (Spears *et al*, 2005) (Zarzhitsky *et al*, 2005) (Spears *et al*, 2004) and potential fields (Esposito & Dunbar, 2006). Those works mostly use a force balance between inter-individual interactions exerting an attractive or repulsive force within the influence range, which might over-constrain the swarm and frequently lead to deadlocks.

Moreover, the computations of relative velocities or accelerations between robots are needed to obtain the magnitude of the force. Regarding the aspect of calculating the movement position of each robot, accuracy and computational efficiency issues will arise.

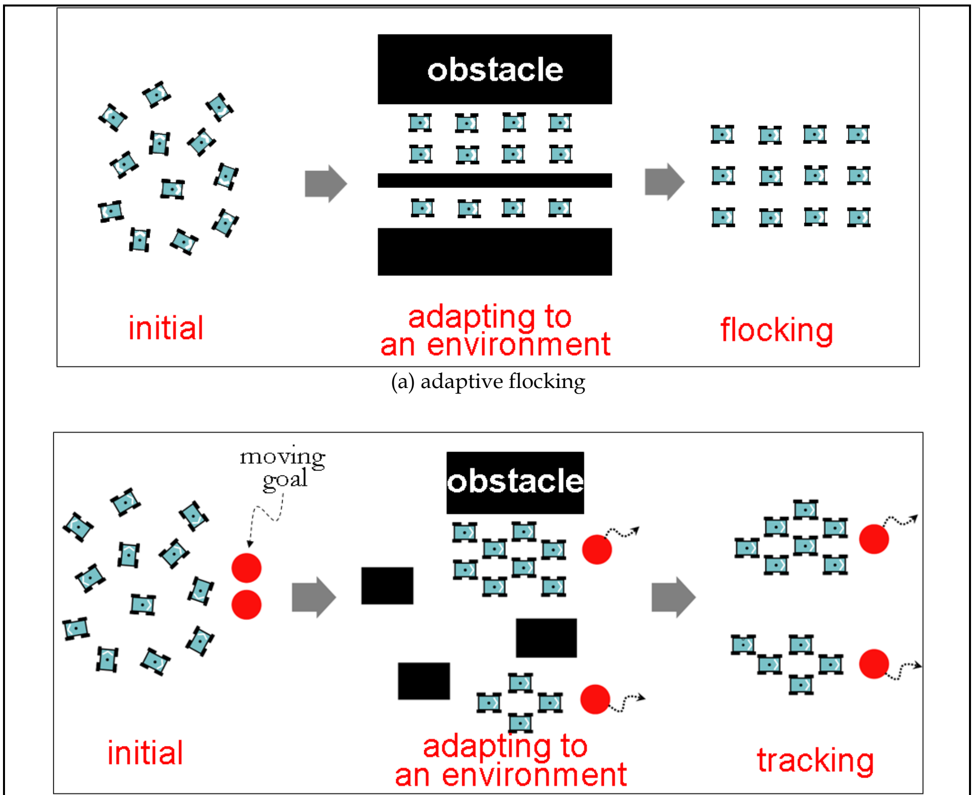
In this paper, from the observation of the habits of schooling fishes, a geometrical motion planning framework locally interacting with two neighbor robots in close proximity is proposed, enabling three neighboring robots to form an equilateral triangle lattice. Based on the local interaction, we develop an adaptive navigation approach that enables a large swarm of autonomous mobile robots to flock through an unknown environment. The proposed approach allows a swarm of robots to split into multiple groups or merge with other groups according to the environmental conditions. Specifically, it is assumed that individual robots are not allowed to have any unique identifier, a pre-determined leader, a common coordinate system, any memory for past decisions and actions, and a direct communication with each other. Given these underlying assumptions, all robots execute the same algorithm and act independently and asynchronously of each other. In spite of such minimal conditions, the above-mentioned potential applications often require a large-scale swarm of robots to navigate toward a certain direction from arbitrary initial positions of the robots in an environment populated with obstacles. For instance, in exploration and search-and-rescue operations, robot swarms need to be dispersed into an unknown area of interest in a uniform spatial density and search for targets. Consequently, the proposed approach provides an efficient yet robust way for robot swarms to self-adjust their shape and size according to the environment conditions. This approach can also be considered as an *ad hoc* mobile networking model whose connectivity must be maintained in a cluttered environment.

The rest of this paper is organized as follows. Section 2 presents the robot model and the statement of the swarm flocking problem. Section 3 describes the basic motion planning of each individual robot locally interacting with neighboring robots. Section 4 presents a collective solution to the swarm flocking problem. Section 5 illustrates how to extend the solution algorithms to the swarm tracking problem. Section 6 provides the results of simulations and discussion. Section 7 draws conclusions.

2. Problem Statement

We consider a swarm of n autonomous mobile robots, where individual robots are denoted respectively by $r_1, r_2, \cdots, r_n$. Each robot is modeled as a point, which freely moves on a two-dimensional plane. It is assumed that the initial distribution of robots is arbitrary and distinct. The robots have no leader and no unique identification numbers. They do not share any common coordinate system, and do not retain any memory of past actions that gives inherently self-stabilizing property[1] (Suzuki & Yamashita 1999). They can detect the positions of other robots within their limited ranges of sensing, but do not have any explicit direct means of communication to each other. Each of the robots executes the same algorithm, but acts independently and asynchronously from other robots. They repeat an endless activation cycle of observation, computation, and motion.

[1] Self-stabilization is the property of a system which, started in an arbitrary state, always converges toward a desired behavior (Dolev, 2000) (Schneider, 1993).



(a) adaptive flocking

(b) adaptive tracking

Fig. 1. Illustration of two flocking control problems

Denote the distance between any two robots r_i and r_j, located respectively at p_i and p_j, as $dist(p_i, p_j)$. Also denote a constant distance as d_u that is finite and greater than zero. Each robot has a limited sensing boundary *SB*. Then r_i detects the positions of other robots, $\{p_1, p_2, \cdots\}$, located within its *SB*, and makes a set of the observed positions O_i obtained with respect to its local coordinate system. From O_i, r_i can select two specific robots r_{s1} and r_{s2}, respectively. We call r_{s1} and r_{s2} the *neighbor* of r_i, and define their positions $\{p_{s1}, p_{s2}\}$ as the *neighbor set* N_i. Given p_i and N_i, *Triangular Configuration* is defined as a set of three distinct positions $\{p_i\ p_{s1}, p_{s2}\}$ denoted by T_i. Next, we can define *Equilateral Configuration* E_i if and only if all the possible distance permutations $dist(p_{\pi(i)}, p_{\pi(j)})$ in T_i are equal to d_u. In this paper, each robot attempts to follow a certain rule to generate E_i from an arbitrary T_i. We formally define each individual robot's behavior as *Local Interaction*, which allows the position of r_i to be maintained to be d_u with N_i at each time toward forming E_i. Now, we can address the following problem of *Adaptive Flocking* for a swarm of robots based on local interactions (see Fig. 1):

- (***Adaptive Flocking***) Given $r_1, \cdots, r_n$ located at arbitrarily distinct positions in a two dimensional plane, how to enable the robots to move toward a stationary or moving goal while adapting to an environment populated with obstacles.

3. Local Interaction

ALGORITHM - 1 LOCAL INTERACTION (code executed by each robot r_i)

```
constant d_u := a uniform distance
Function φ_interaction(O_i, p_i)
    (p_ct,x, p_ct,y) := centroid( p_i, {p_s1, p_s2} )
    ϕ := angle between p_s1 p_s2 and r_i's local horizontal axis
    p_ti,x := p_ct,x + d_u cos(ϕ + π/2) / √3
    p_ti,y := p_ct,y + d_u cos(ϕ + π/2) / √3
    p_ti := ( p_ti,x , p_ti,y )
```

Local geometric shapes of a school of tuna are known to form a diamond shape (Stocker, 1999), whereby tunas exhibit the following schooling behaviors: maintenance, partition, and unification. Similarly, local interaction for a swarm of robots in this paper is to generate an equilateral triangular lattice. This section explains how the local interaction is established among three neighboring robots.

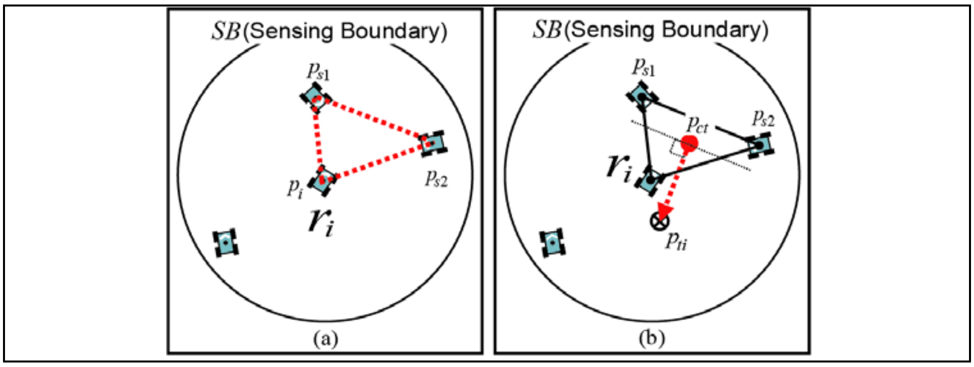


Fig. 2. Illustration of local interaction ((a) triangular configuration, (b) target computation))

As presented in ALGORITHM-1, the algorithm consists of a function $\varphi_{interaction}$ whose arguments are p_i and N_i at each activation step. Consider any robot r_i and its two neighbors r_{s1} and r_{s2} located within its SB. As shown in Fig. 2-(a), three robots are configured into T_i whose vertices are p_i, p_{s1} and p_{s2}, respectively. First, r_i finds the centroid of the triangle $\Delta\, p_i p_{s1} p_{s2}$, denoted by p_{ct}, with respect to its local coordinates, and measures the angle ϕ between the line connecting the two neighbors and r_i's horizontal horizontal axis. Using p_{ct} and ϕ, r_i calculates the target point p_{ti} as illustrated in Fig. 2-(b).

Each robot computes the target point by their current observation of neighboring robots. Intuitively, under ALGORITHM-1, r_i may maintain d_u with its two neighbors at each time. In other words, each robot attempts to form an isosceles triangle for N_i at each time, and by repeatedly doing this, three robots configure themselves into E_i .

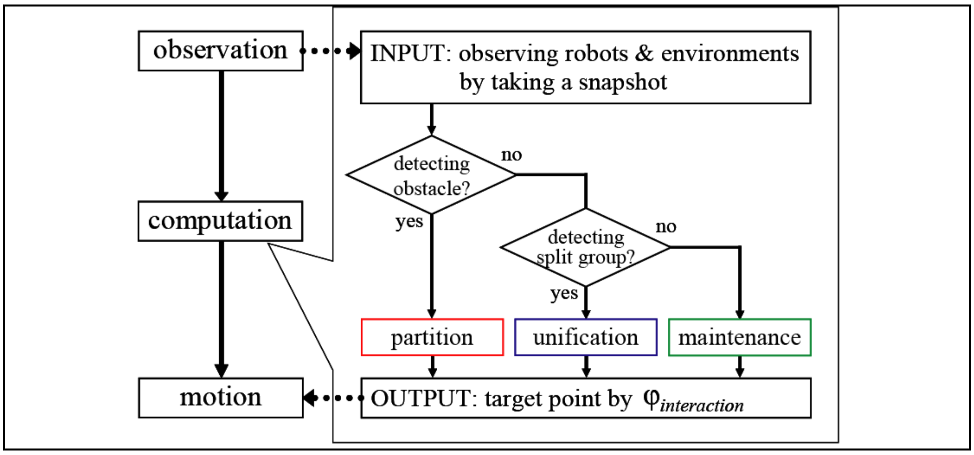


Fig. 3. Adaptive flocking flowchart

4. Adaptive Flocking Algorithm

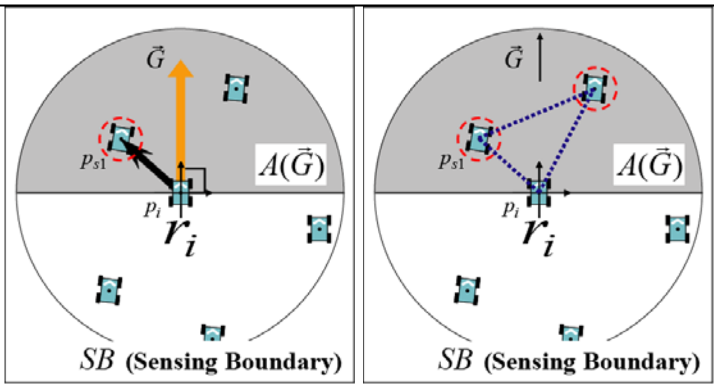
4.1 Architecture of Adaptive Flocking

The adaptive flocking problem addressed in Section 2 can be decomposed into three sub-problems as illustrated in Fig. 3, each of which is solved based on the same local interaction (see Section 3).

- *Maintenance*: Given that robots located at arbitrarily distinct positions, how to enable the robots to flock in a single swarm.
- *Partition*: Given that an environmental constraint is detected, how to enable a swarm to split into multiple smaller swarms adapting to the environment.
- *Unification*: Given that multiple swarms exist in close proximity, how to enable them to merge into a single swarm.

As illustrated in Fig. 3, the input of the algorithm for each time instant is O_i and the environment information with respect to the local coordinate system of each robot. The output is p_{ti} computed by $\varphi_{interaction}$. At each time, r_i can either be idle or execute their algorithm, repeating recursive activation at each cycle. At each cycle, each robot computes their movement positions (computation), based on the positions of other robots (observation), and moves toward the computed positions (motion). Through this activation cycle, when the robot finds any geographical constraint within its *SB*, the robot executes the *partition algorithm* to adapt its position to the constraint. On the other hand, when the robot finds no geographical constraint, but observes any robot around the outside of its group, the robot executes the *unification algorithm*. Otherwise, the robot basically executes the *maintenance algorithm* while navigating toward a goal.

4.2 Team Maintenance



(a) 1st neighbor selection (b) 2nd neighbor selection

Fig. 4. Illustration of team maintenance

The first problem is how to maintain a uniform interval among individual robots while navigating. This enables the robots to form a multitude of equilateral triangle lattices. Each robot adjusts $\vec{G}$, termed the goal direction, with respect to its local coordinates and computes O_i at the time t. As illustrated in Fig. 4-(a), let $A(\vec{G})$ denote the area of goal direction defined within the robot's SB. Next, each robot checks whether there exists a neighbor in $A(\vec{G})$. If multiple neighbors exist, r_i selects the first neighbor r_{s1} located the shortest distance away from p_i and defines its position as p_{s1}. Otherwise, r_i spots a virtual point p_v located an adequate distance d_v away from p_i along $\vec{G}$, defined as p_{s1} . As shown in Fig. 4-(b), the second neighbor r_{s2} is selected such that the total distance from p_{s1} to p_i passing through p_{s2} is minimized. As a result, p_{ti} can be obtained by $\varphi_{interaction}$ in ALGORITHM-1.

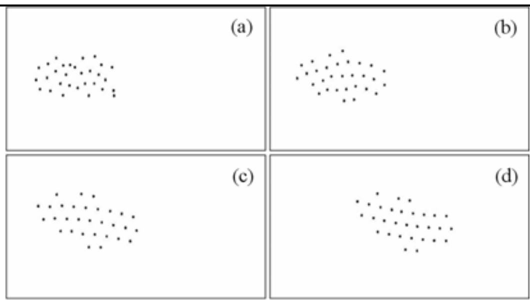
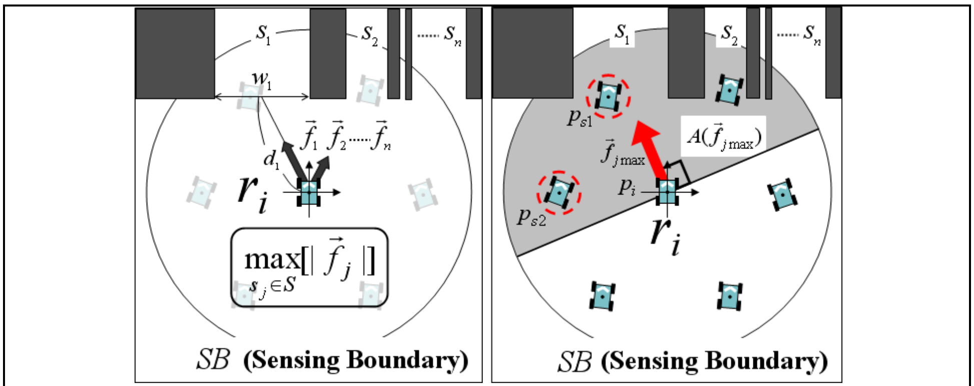


Fig. 5. Simulation for maintenance algorithm ((a) initial distribution, (b) 2 sec. (c) 4 sec. (d) 11 sec.)

Fig. 5 shows the simulation results of maintenance algorithm with 30 robots under no environmental constraints. Initially, robots are arbitrarily located on the two-dimensional plane. As shown in Figs. 5-(b) and (c), each robot generates its geometric configuration with their neighbors while moving toward a goal. Fig. 5-(d) illustrates that robots maintain a single swarm while navigating. Once the target is detected by any of the robots closest to the goal, the swarm could navigate toward the goal through individual local interactions.

4.3 Team Partition



(a) favorite vector (b) neighbor selection

Fig. 6. Illustration of team partition

When a swarm of robots detects an obstacle in its path, each robot is required to determine its direction toward the goal avoiding the obstacle. In this paper, each robot determines their direction by using the relative degree of attraction of the passageway (Halliday *et al.*, 2007), termed the favorite vector $\vec{f}$, whose magnitude is given by

$$|\vec{f}_j| = |w_j / d_j^2|. \tag{1}$$

In Fig. 6-(a), s_j denotes the passageway with width w_j, and d_j denotes the distance between the center of w_j and p_i. Note that if r_i can not exactly measure w_j beyond its *SB*, w_j is set to the maximum value of *SB*. Now the passageways can be represented by a set of favorite vectors $\{\vec{f}_j | 1 \le j \le n\}$ and then r_i selects the maximum magnitude of $\vec{f}_j$ denoted as $|\vec{f}_j|_{\max}$. As shown in Fig. 6-(b), r_i defines a maximum favorite area $A(\vec{f}_{j\max})$ based on the direction of $|\vec{f}_j|_{\max}$ within its *SB*. Next, r_i checks whether there exists a neighbor in $A(\vec{f}_{j\max})$. If neighbors are found, r_i selects r_{s1} located the shortest distance away from itself to define p_{s1}. Otherwise, r_i spots a virtual point p_v located at an adequate distance d_v in the direction of $|\vec{f}_j|_{\max}$ to define p_{s1}. Finally r_{s2} is selected such that the total distance from p_{s1} to p_i passing through p_{s2} is minimized. As a result, p_{ti} can be obtained by $\varphi_{interaction}$ in ALGORITHM-1.

In Fig. 7, there existed three passageways in the environment. Based on the proposed algorithm, robots could be split into three smaller groups while maintaining the local geometric configuration. Through the local interactions, the rest of the robots could naturally adapt to an environment by just following their neighbors moving ahead toward the goal.

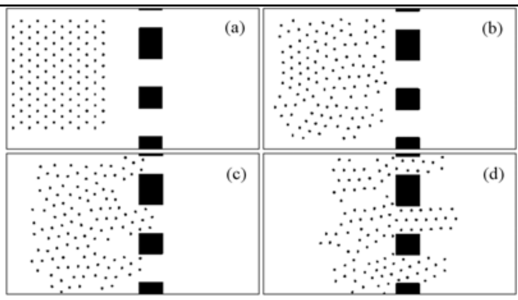
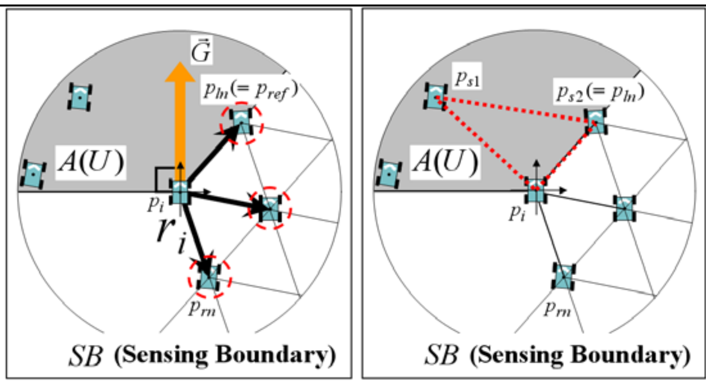


Fig. 7. Simulation for partition algorithm ((a) initial distribution, (b) 5 sec. (c) 9 sec. (d) 18 sec.)

4.4 Team Unification



(a) unification area (b) neighbor selection

Fig. 8. Illustration of team unification

In order to enable the multiple swarms in close proximity to merge into a single swarm, r_i adjusts $\vec{G}$ with respect to its local coordinates and defines the position set of robots D_u located within the range of d_u. Let $ang(\vec{m},\vec{n})$ be an angle between two arbitrary vectors $\vec{m}$ and $\vec{n}$. As shown in Fig. 8-(a), r_i computes $ang(\vec{G},\overrightarrow{p_i p_{uk}})$, where $\overrightarrow{p_i p_{uk}}$ is the vectors starting from p_i to p_{uk} of D_u, and defines the neighbor position p_{ref} that gives the minimum $ang(\vec{G},\overrightarrow{p_i p_{uk}})$ between $\vec{G}$ and $\overrightarrow{p_i p_{uk}}$. Starting from $\overrightarrow{p_i p_{ref}}$, r_i checks whether there exists the neighbor position p_{ul} which belongs to D_u within the area obtained by rotating $\overrightarrow{p_i p_{ref}}$ 60 degrees clockwise. If there exists p_{ul}, r_i finds another neighbor position p_{um} using the same method starting from $\overrightarrow{p_i p_{um}}$. Unless p_{ul} exists, r_i defines p_{ref} as p_{rn}. Similarly, r_i can decide the neighbor position p_{ln} while rotating 60 degrees counter clockwise from $\overrightarrow{p_i p_{ref}}$. The two positions, denoted as p_{rn} and p_{ln} , are located farthest in the right-hand or left-hand direction of $\overrightarrow{p_i p_{ref}}$, respectively. As illustrated in Fig. 8-(b), a unification area $A(U)$ is defined as the common area between $A(\vec{G})$ in SB and the rest of the area in SB, where no element of D_u exists. Then, r_i defines a set of robots in $A(U)$ and selects the first neighbor r_{s1} located the shortest distance away from p_i in $A(U)$. The second neighbor position is defined such that the total distance from p_{s1} to p_i can be minimized through either p_{rn} or p_{ln} . As a result, p_{ti} can be obtained by $\varphi_{interaction}$ in ALGORITHM-1. Fig. 9 demonstrates how two separate groups of 120 robots merge into one while maintaining the local geometrical configuration.

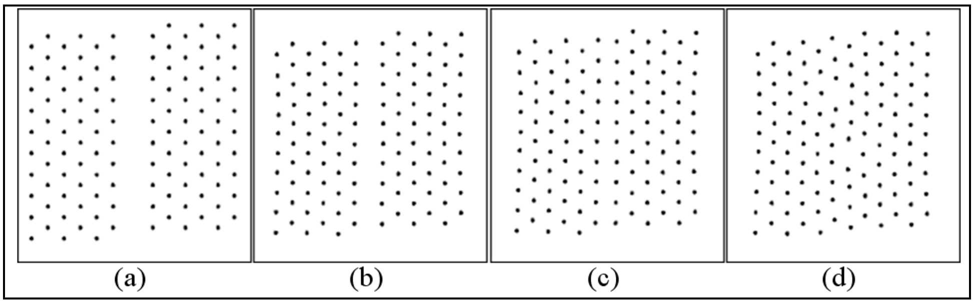


Fig.9. Simulation for unification algorithm ((a) initial distribution, (b) 5 sec. (c) 14 sec. (d) 20 sec.)

5. Adaptive Tracking Algorithm

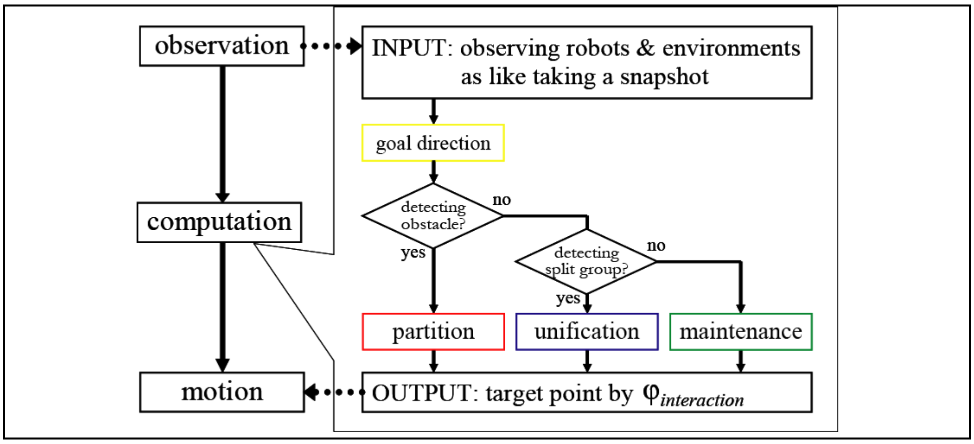
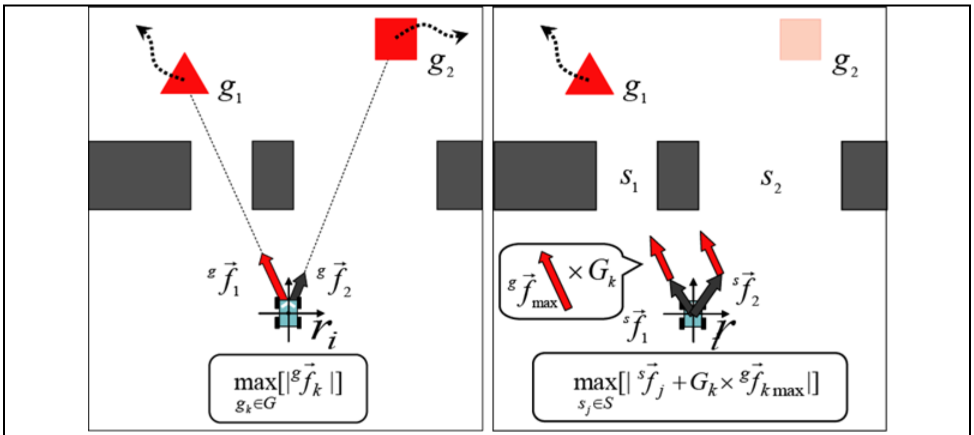


Fig. 10. Adaptive tracking flowchart

This section introduces a straightforward extension of adaptive flocking to a more sophisticated example of swarm behavior that enables groups of robots to follow multiple moving goals while adaptively navigating through an environment populated with obstacles. Fig. 10 shows the flowchart of this adaptive tracking application. Under the same activation cycle as described in Section 4, each robot first identifies the goal(s) in its *SB* and selects a single goal to track. After adjusting the *goal direction*, when the robot finds the geographical constraint within its *SB*, the robot executes the *partition algorithm* to adapt its position to the constraint. If the robot finds no constraint, but observes any robot around the outside of its group, the robot executes the *unification algorithm*. Otherwise, the robot basically executes the *maintenance algorithm* while navigating toward the selected goal. Notice that the adaptive tracking differs from the adaptive flocking in computation of the goal direction detailed below. Specifically, the partition in the tracking is to enable a single swarm to be divided into smaller groups according to an environmental constraint and/or selected goal.



(a) computation of goal favorite vectors (b) compuation of navigation direction

Fig. 11. Illustrating direction selection in adaptive tracking

In Fig. 11, similar to Eq. (1), the favorite vector for the passageway is defined as ${}^s\vec{f}_j$. Likewise, the tracking goal is defined as ${}^g\vec{f}_k$. Assuming that one of the goals g_k is located some distance d_k away from p_i, the magnitude of the favorite vector ${}^g\vec{f}_k$ for the goal is given by

$$|{}^g\vec{f}_k| = |1/d_k^2|\,. \tag{2}$$

Here, it is assumed that the set of multiple moving goals *GS*, $\{{}^g\vec{f}_k \,|\, 1 \le k \le n\}$, has the same priority across the respective goals. From *GS*, r_i selects a favorite vector with the maximum magnitude denoted as $|{}^g\vec{f}_k|_{\max}$. As described in the Subsection 4.3 (see Fig. 6-(b)), r_i defines the maximum favorite area $A({}^g\vec{f}_{k\max})$ and selects the neighbors within $A({}^g\vec{f}_{k\max})$.

Next, let us consider the case that r_i observes both the goals and passageways. As shown in Fig. 11-(a), r_i first defines the favorite vectors of the observed goals ${}^g\vec{f}_k$, and then selects g_k with $|{}^g\vec{f}_k|_{\max}$. With respect to the selected goal, as seen in Fig. 11-(b), r_i selects s_j based on the following measure

$$\max_{s_j \in S}\left[|{}^{s}\vec{f}_j + G_k \times {}^{g}\vec{f}_{k\max}|\right] \quad (3)$$

where G_k indicates a weighting coefficient in order to upset the balance between ${}^{s}\vec{f}_j$ and ${}^{g}\vec{f}_k$. Similar to the previous approach, r_i defines $A({}^{s}\vec{f}_{k\max})$ where the first neighbor is selected.

6. Simulation Results and Discussion

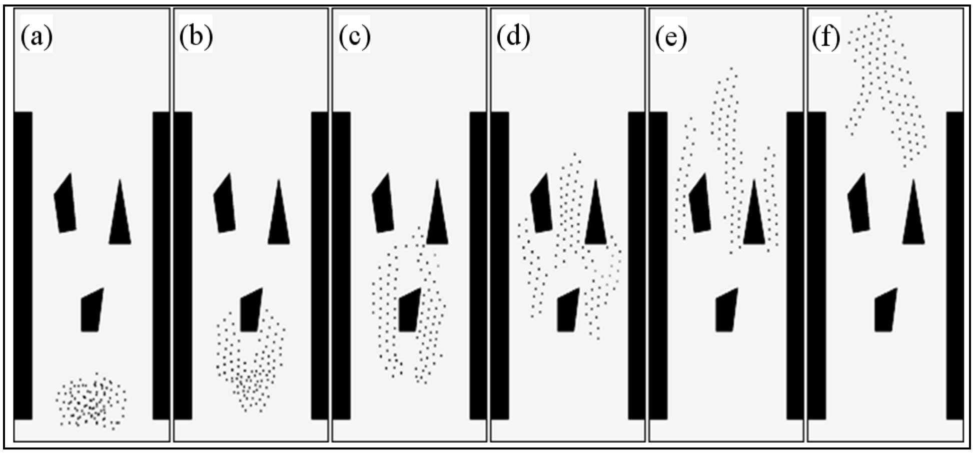


Fig. 12. Simulation results of adaptive flocking toward a stationary goal

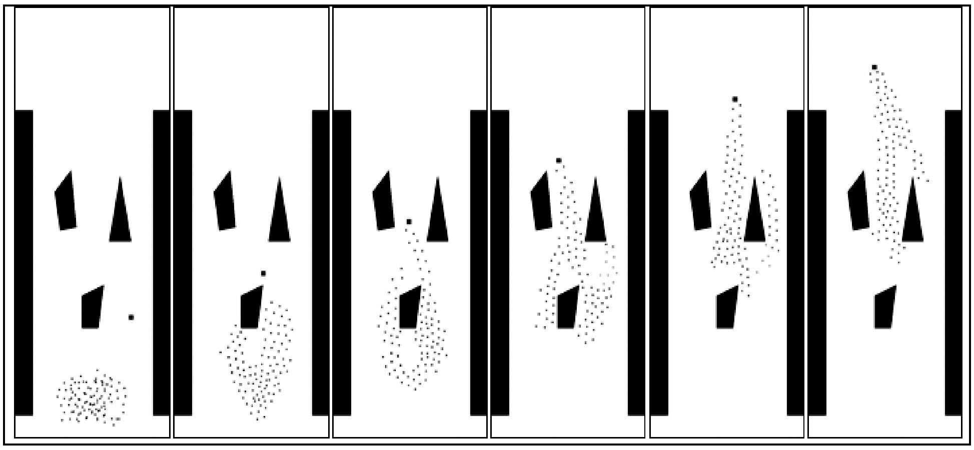


Fig. 13. Simulation results of adaptive tracking toward a moving goal

To verify the proposed flocking and tracking algorithms, simulations are performed with a swarm of 100 robots. We set the distance d_v between p_v and p_i to 1.2 times longer than d_u

and the range of SB to 3.5 times longer than d_u . Moreover, in the tracking simulations, G_k was set to 10. The first simulation demonstrates how a swarm of robots adaptively flocks in an unknown environment populated with obstacles. In Fig. 12, the swarm navigates toward a stationary goal located at the upper center point. On the way to the goal, some of the robots detect an obstacle that forces the swarm split into two groups in Fig. 12-(b). The rest of the robots can just follow their neighbors moving ahead toward the goal. After being split into two groups, each group maintains the geometric configuration while navigating in Fig. 12-(c). Note that the robots that could not identify the obstacle just follow the moving direction of preceding robots. Figs. 12-(d) and (e) show that two groups are merged and/or split again into smaller groups due to the next obstacles. In Fig. 12-(f), the robots successfully pass through the environment.

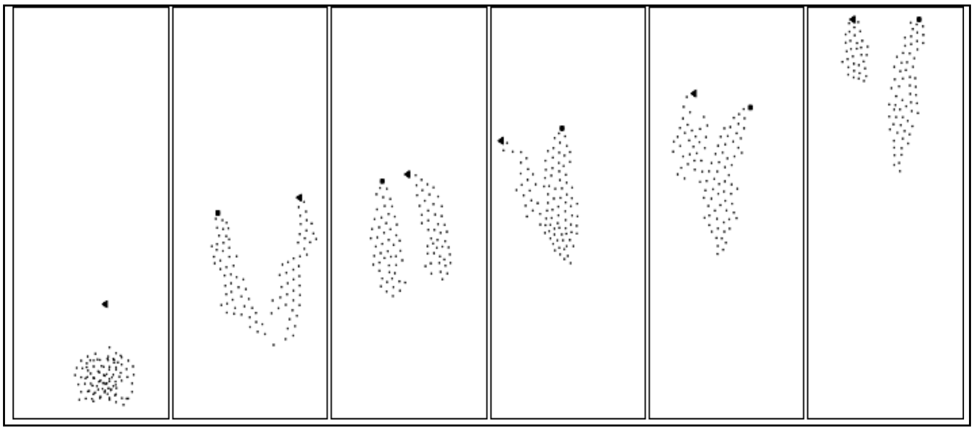


Fig. 14. Simulation results of two moving goals tracking in free space

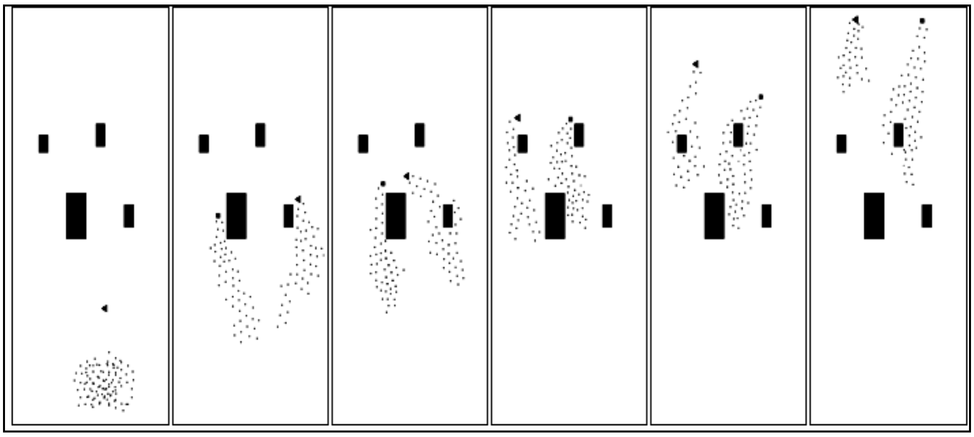


Fig. 15. Simulation results of tracking two moving goals in a geographically-constrained environmental constraint

The next simulation results seen in Fig. 13 present the snapshots for tracking of a moving goal represented by the square. As the goal moves, the swarm starts to move. It can be observed that the snapshots of Fig. 13 differ from those of Fig. 12, since ${}^{s}\vec{f}_j$ varies in accordance with $\vec{G}$ detected at each time.

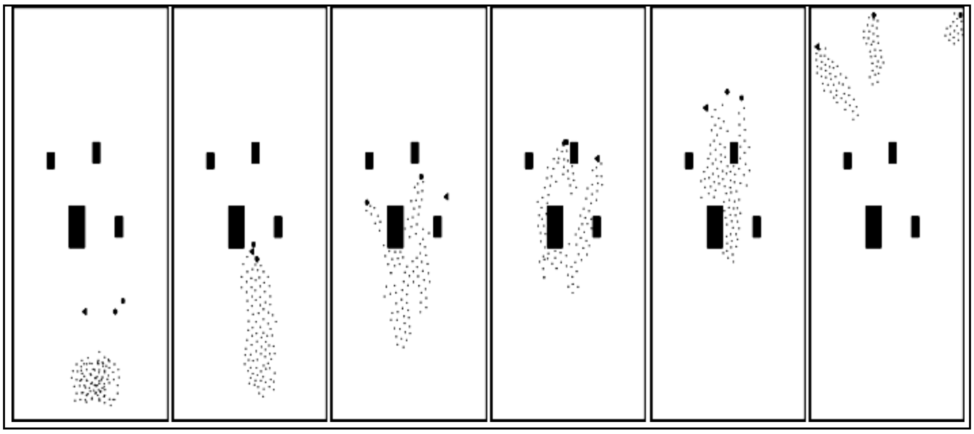


Fig. 16. Simulation results of tracking three moving goals in a geographically-constrained environmental constraint

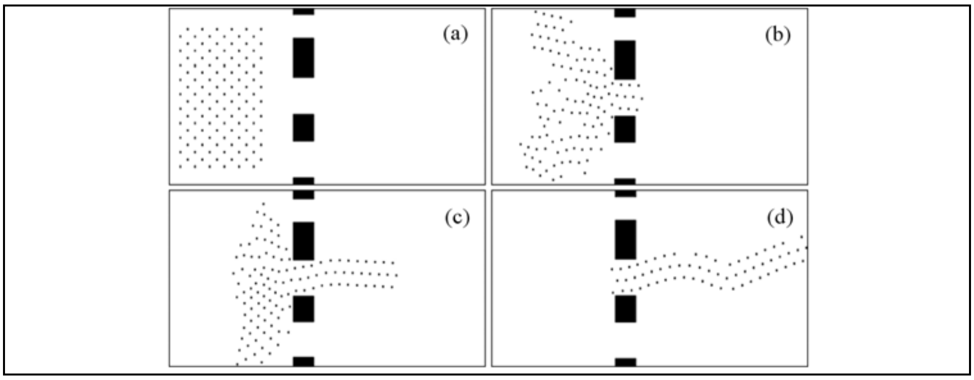


Fig. 17. Simulation for flocking without partition capability ((a) initial distribution, (b) 13 sec. (c) 52 sec. (d) 148 sec.)

Figs. 14 and 15 present the snapshots that the same swarm tracks two moving goals having different velocities represented by the square and the triangle, respectively. The simulation conditions are the same, but Fig. 15 is carried out in the environment populated with obstacles. In addition, Fig. 16 shows how the swarm tracks three moving goals in the same environment. It can be observed that the swarm behavior of each case differs as expected.

In Fig. 17, we investigate the swarm behavior when the partition capability is not available. It took about 150 seconds to pass through the passageway. In the simulation result of Fig. 7, it took about 50 seconds with the same velocity and d_u. From this, it is evident that the partition provides a swarm with an efficient navigation capability in an obstacle-cluttered environment. Likewise, unless the robots have the unification capability, they may separately perform a common task after being divided as presented in Fig. 18. The capability of unification can be used to make performing a certain task easier, which may not be completed by an insufficient number of robots.

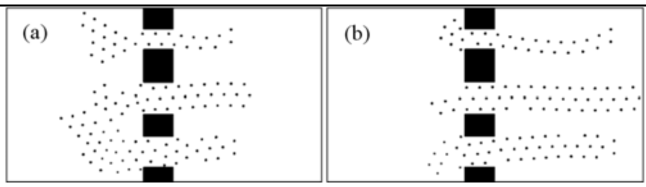


Fig. 18. Simulation for flocking without unification capability ((a) 28 sec., (b) 40 sec.)

We believe that our algorithms work well under real world conditions, but several issues remain to be addressed. It would be interesting to verify (1) if the performance of the algorithms is sensitive to measurement errors caused by unreliable sensors, or (2) if the algorithms can be extended to three dimensional space. The algorithms rely on the fact that robots can identify other robots and distinguish them from various objects using, for instance, sonar reading (Lee & Chong, 2006) or infrared sensor reading (Spears *et al*, 2004). This important engineering issue is left for future work. Regarding using explicit direct communications, it also suffers from limited bandwidth, range, and interferences. Moreover, it is necessary for robots to use *a priori* knowledge such as identifiers or global coordinates (Lam & Liu, 2006) (Nembrini *et al*, 2002). We are currently studying the relation between the robot model (or capabilities) and different communication (or interaction) models.

7. Conclusion

In this paper, we presented a decentralized algorithm of adaptive flocking and tracking, enabling a swarm of autonomous mobile robots to navigate toward achieving a mission while adapting to an unknown environment. Through local interactions by observing the position of the neighboring robots, the swarm could maintain a uniform distance between individual robots, and adapt its direction of heading and geometric shape. We verified the effectiveness of the proposed strategy using our in-house simulator. The simulation results clearly demonstrated that the proposed flocking and tracking are a simple and efficient approach to autonomous navigation for robot swarms in a cluttered environment by repeating the process of splitting and merging of groups passing through multiple narrow passageways. In practice, this approach is expected to be used in applications such as odor localization, search-and-rescue, and *ad hoc* mobile networking.

Finally, we emphasize several points that highlight unique features of our approach. First, an equilateral triangle lattice is built with a partially connected mesh topology. Among all the possible types of regular polygons, the equilateral triangle lattices can reduce the computational burden and become less influenced by other robots, due to the limited number of neighbors, and be highly scalable. Secondly, the proposed local interaction is computationally efficient, since each robot utilizes only position information of other robots. Thirdly, our approach eliminates such major assumptions as robot identifiers, common coordinates, global orientation, and direct communication. More specifically, robots compute the target position without requiring memories of past actions or states, helping cope with transient errors.

8. References

Reynolds, C. W. (1987). Flocks, herds, and schools: a distributed behavioral model, *Computer Graphics*, Vol. 21, No. 4, 25-34

Egerstedt, M. & Hu, X. (2001). Formation constrained multi-agent control, *IEEE Transactions on Robotics and Automation*, Vol.17, No.6, 947-951

Burgard, W.; Moors, M. & Stachniss, C. & Schneider, F. (2005). Coordinated multi-robot exploration, *IEEE Transactions on Robotics and Automation*, Vol.20, No.3, 120-145

Baldassarre, G.; Trianni, V. & Bonani, M. & Mondada, F. & Dorigo, M. & Nolfi S. (2007). Self-organized coordinated motion in groups of physically connected robots, *IEEE Transactions on Systems, Man, and Cybernetics - Part B*. Vol.37, No.1, 224-239

Shimizu, M.; Mori, T. & Ishiguro, A. (2006). A development of a modular robot that enables adaptive reconfiguration, *Proc. IEEE/RSJ International Conference on Intelligent Robots and Systems*, 174-179, IEEE

Folino, G. & Spezzano, G. (2002). An adaptive flocking algorithm for spatial clustering, In: *Parallel Problem Solving from Nature* (*LNCS)*, Vol.2439, 924-933, Springer Berlin, 978-3-540-44139-7

Ogren, P. & Leonard, N. E. (2005). A convergent dynamic window approach to obstacle avoidance, *IEEE Transactions on Robotics and Automation*, Vol.21, No.2, 188-195

Balch, T. & Hybinette, M. (2000). Social potentials for scalable multi-robot formations, *Proc. IEEE International Conference on Robotics and Automation*, 73-80, IEEE

Lee, G. & Chong, N. Y. (2006). Decentralized formation control for a team of anonymous mobile robots, Proc. 6th Asian Control Conference, 971-976, 979-15017-0

Spears, W.; Spears, D. & Hamann, J. & Heil, R. (2004). Distributed, physics-based control of swarms of vehicles, *Autonomous Robots*, Vol.17, No.2-3, 137-162

Spears, D.; Kerr, W. & Spears, W. (2006). Physics-based robot swarms for coverage problems, *International Journal on Intelligent Control and Systems*, Vol.11, No.3, 24-140

Esposito, J. M. & Dunbar, T. W. (2006). Maintaining wireless connectivity constraints for swarms in the presence of obstacles, *Proc. IEEE International Conference on Robotics and Automation*, 946-951, IEEE

Zarzhitsky, D.; Spears, D. & Spears, W. (2005). Distributed robotics approach to chemical plume tracing, *Proc. IEEE/RSJ International Conference on Intelligent Robots and Systems*, 1-6, IEEE

Sahin, E. (2005). Swarm robotics: from sources of inspiration to domains of application, In: *Swarm Robotics* (*LNCS)*, Vol.3342, 10-20, Springer Berlin, 978-3-540-24296-3

Lam, M. & Liu, Y. (2006). ISOGIRD: an efficient algorithm for coverage enhancement in mobile sensor networks, *Proc. IEEE/RSJ International Conference on Intelligent Robots and Systems*, 1458-1463, IEEE

Nembrini, J.; Winfield, A. & Melhuish, C. (2002). Minimalist coherent swarming of wireless networked autonomous mobile robots, Proc. 7th International Conference on Simulation of Adaptive Behavior, 373-382, IEEE

Suzuki, I. & Yamashita, M. (1999). Distributed anonymous mobile robot: formation of geometric patterns, *SIAM Journal of Computing*, Vol.28, No.4, 1347-1363

Dolev, S. (2000). Self-Stabilization, *MIT Press*, 978-0262041782

Schneider, M. (1993). Self-stabilization, *ACM Computing Survey*, Vol.25, No.1, 45-67

Wilson, E. O. (2000). *Sociobiology: the new synthesis*, Belknap Press, 978-067400235-7

Stocker, S. (1999). Models for tuna school formation, *Mathematical Biosciences*, Vol.156, 167-190

Halliday, D.; Resnick, R. & Walker, J. (2007). *Fundamentals of physics*, John Wiley & Sons Inc, 8th ed., 978-0471758013