5

Adaptive Navigation Control for Swarms of Autonomous Mobile Robots

Yasuhiro Nishimura[1], Geunho Lee[1], Nak Young Chong[1], Sang Hoon Ji[2] and Young-Jo Cho[3]

[1]*Japan Advanced Institute of Science and Technology,*
[2]*Korea Institute of Industrial Technology,*
[3]*Electronics and Telecommunications Research Institute*
[1]*Japan*
[2,3]*Korea*

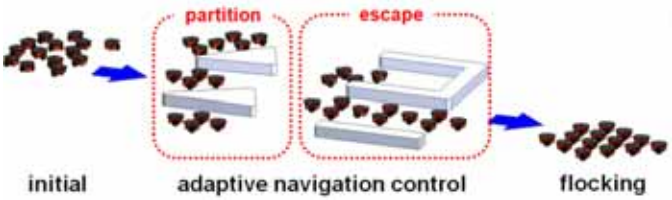
1. Introduction

Deploying a large number of resource-constrained mobile robots performing a common group task may offer many advantages in efficiency, costs per system, and fault-tolerance (Sahin, 2005). Therefore, robot swarms are expected to perform missions in a wide variety of applications such as environment and habitat monitoring, exploration, odor localization, medical service, and search-and-rescue. In order to perform the above-mentioned tasks successfully, one of the most important concerns is how to enable swarms of simple robots to autonomously navigate toward a specified destination in the presence of obstacles and dead-end passageways as seen in Fig. 1. From the standpoint of the decentralized coordination, the motions of individual robots need to be controlled to support coordinated collective behavior.

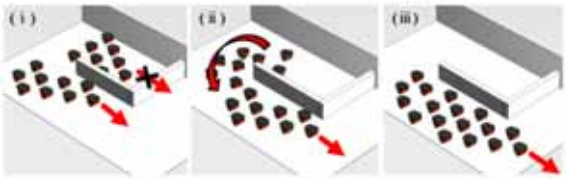
We address the coordinated navigation of a swarm of mobile robots through a cluttered environment without hitting obstacles and being trapped in dead-end passageways. Our study is motivated by the observation that schools of fish exhibit emergent group behavior. For instance, when schools of fish are faced with obstacles, they can split themselves into a plurality of smaller groups to avoid collision and then merge into a single group after passing around the obstacles (Wilson, 1976). It is also worth noting that a group of fish facing a dead end can get out of the area.

Based on the observation of schooling behavior in fish, this work aims to present a novel adaptive group behavior, enabling large-scale robot swarms with limited sensing capabilities to navigate toward a goal that is visible only to a limited number of robots. In particular, the coordinated navigation is achieved without using any leader, identifiers, common coordinate system, and explicit communication. Under such a minimal robot model, the adaptive navigation scheme exploits the geometric local interaction which allows three neighboring robots to form an equilateral triangle. Specifically, the proposed algorithm allows robot swarms to 1) navigate while maintaining equilateral triangular lattices, 2) split themselves into multiple groups while maintaining a uniform distance of each other, 3) merge into a single group while maintaining a uniform distance of each other,

and 4) escape from any dead-end passageways. During the adaptive navigation process, all robots execute the same algorithm and act independently and asynchronously of each other. Given any arbitrary initial positions, a large-scale swarm of robots is required to navigate toward a goal position in an environment while locally interacting with other robots. The basic necessities for the proposed solution are argued as follows. First, the robots can self-control their travel direction according to environmental conditions, leading to enhancing autonomy of their behavior. Secondly, by being split into multiple groups or re-united into a single swarm, the robots can self-adjust its size and shape depending on the conditions. By the capabilities above, robots have the emergent capability to maximize adaptability to operate in uncertain environments. Thirdly, the coordinated navigation of multiple robots in an equilateral triangle formation reduces a potential traffic jam and stragglers.



(a) adaptive navigation problem



(b) escape function

Fig. 1. Concept of adaptive navigation by autonomous mobile robot swarms

Consequently, the proposed adaptive navigation provides a cost-effective way to allow for an increase in efficiency and autonomy of group navigation in a highly cluttered environment. What is important from a practical standpoint is that the swarm flocking is considered as a good *ad hoc* networking model whose connectivity must be maintained while moving. In particular, maintaining the uniform distance enables the model to optimize efficient energy consumption in routing protocols (Fowler, 2001)(Lyengar *et al.*, 2005). This networking model can potentially be used in application examples such as exploration and search-and-rescue. This navigation can be further applied to swarms of unmanned vehicles and sensors performing autonomous operations such as capturing and transporting toxic and hazardous substances. We describe our algorithm in detail, and perform extensive simulations to demonstrate that a swarm of robots can navigate toward a specified destination while adapting to unknown environmental conditions in a scalable manner.

The rest of this paper is organized as follows. Section 2 introduces a brief description of research related to swarming and flocking and sheds light on our motivation. Section 3 presents the robot model and the formal definitions of the adaptive navigation problem. Section 4 describes the fundamental motion control of each robot locally interacting with

their neighboring robots, leading to forming an equilateral triangle lattice. Section 5 gives the solution scheme of the adaptive navigation. Section 6 demonstrates the validity and applicability of the proposed scheme through extensive simulations. Section 7 draws conclusions.

2. Background

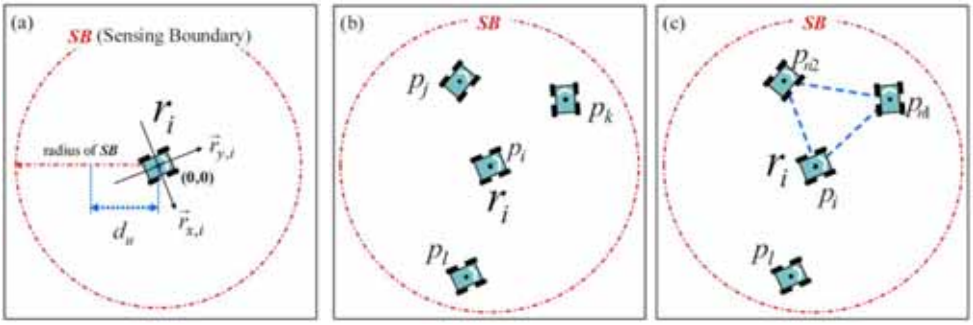
Wireless network-enabled mobile robotic sensors have been increasingly popular over the recent years (Yicka *et al.*, 2008). Such robotic sensors dispersing themselves into an area can be used for search-and-rescue and exploration applications by filling the area of interest and/or establishing an *ad hoc* network. To achieve a desired level of self-deployment of robotic sensors, many prior studies have attempted to use decentralized approaches in self-configuration (Lee & Chong, 2009)(Shucker *et al.*, 2008)(Spears *et al.*, 2004), pattern generation (Lee & Chong, 2009-b)(Ikemoto *et al.*, 2005), and navigation (Gu & Hu, 2010)(Lee & Chong, 2008)(Olfati-Saber, 2006). In particular, we have witnessed a great interest in distributed navigation control that enables a large number of robots to navigate from an initial position toward a desired destination without human intervention.

Recently, many navigation control studies have been reported in the field of swarm robotics, where the decentralized navigation controls are mainly based on interactions between individual robots mostly inspired by evidence from biological systems (*e.g.*, fish schools or bird flocks) or natural phenomena (*e.g.*, liquid diffusion). The navigation control can be further divided into biological emergence (Folino & Spezzano, 2002)(Reynolds, 1987), behavior-based (Lee & Chong, 2008)(Ogren & Leonard, 2005)(Balch & Hybinette, 2000), and virtual physics-based (Esposito & Dunbar, 2006)(Zarzhitsky *et al.*, 2005)(Spears *et al.*, 2006) approaches. Specifically, the behavior-based and virtual physics-based approaches are related to the use of such physical phenomena as gravitational forces (Zarzhitsky *et al.*, 2005)(Spears *et al.*, 2006) and potential fields (Esposito & Dunbar, 2006). Those works mostly use some sort of force balance between inter-individual interactions exerting an attractive or repulsive force on each other. This is mainly because the force-based interaction rules are considered simple yet effective, and provide an intuitive understanding on individual behavior. However, the computation of relative velocities or accelerations between robots is needed to obtain the magnitude of the interacting force.

Regarding the aspect of calculating the movement position of each robot, accuracy and computational efficiency issues have been attracted. In practice, many works on robot swarms use sensor-rich information and explicit means of communication. Note that if any means of communication would be employed, robots need to identify with each other or use a global coordinate or positioning system (Correll *et al.*, 2000)(Lam & Liu, 2006)(Nembrini *et al.*, 2002). In this paper, we attempt to achieve adaptive navigation without taking advantage of rich computational capabilities and communication. This will allow us to develop robot systems in simple, robust, and non-costly ways. A subset of this work was reported in (Lee & Chong, 2008) which provided mobile robot swarms with basic navigation and adaptation capabilities. The main objective of this paper is to present a completely new and general adaptive navigation coordination scheme assuming a more complicated arena with dead-end passageways. Specifically, we highlight the simplicity and intuition of the self-escape capability without incorporating a combination of sophisticated algorithms.

3. Problem statement

3.1 Robot model and notations



(a) r_i's local coordinate (b) observation set O_i (c) triangular configuration T_i

Fig. 2. Illustration of definition and notation

In this work, we consider a swarm of mobile robots denoted by $r_1,\cdots,r_n$. It is assumed that an initial distribution of all robots is arbitrary and their positions are distinct. Each robot autonomously moves on a two-dimensional plane. Robots have no leader and no identifiers. They do not share any common coordinate system. Due to a limited observation range, each robot can detect the positions of other robots only within its line-of-sight. In addition, robots are not allowed to communicate explicitly with each other.

Next, as illustrated in Fig. 2-(a), let's consider a robot r_i with local coordinate system $\vec{r}_{x,i}$ and $\vec{r}_{y,i}$. The robot's heading direction $\vec{r}_{y,i}$ is defined as the vertical axis of r_i's coordinate system. It is straightforward to determine the horizontal axis $\vec{r}_{x,i}$ by rotating $\vec{r}_{y,i}$ 90 degrees clockwise. The position of r_i is denoted by p_i. Note that p_i is (0, 0) with respect to r_i's local coordinate system. The line segment $\overline{p_i p_j}$ is defined as a straight line between p_i and p_j occupied by another robot r_j. The distance between p_i and p_j is defined as $\mathrm{dist}(p_i,p_j)$. In particular, the desired inter-robot distance between r_i and r_j is denoted by d_u. Moreover, $\mathrm{ang}(\vec{m}_i,\vec{n}_i)$ denote the angle between two arbitrary vectors $\vec{m}_i$ and $\vec{n}_i$.

As shown in Fig. 2-(b), r_i detects the positions p_j, p_k and p_l of other robots located within its sensing boundary SB, yielding a set of the positions $O_i(=\{p_j,p_k,p_l\})$ with respect to its local coordinate system. When r_i selects two robots r_{n1} and r_{n2} within its SB, we call r_{n1} and r_{n2} the neighbors of r_i, and their position set $\{p_{n1},p_{n2}\}$ is denoted by N_i as illustrated in Fig. 2-(c). Given p_i and N_i, a set of three distinct positions $\{p_i,p_{n1},p_{n2}\}$ with respect to r_i is called the triangular configuration $\mathbb{T}_i$, namely $\{p_i,p_{n1},p_{n2}\}$. We further define the equilateral triangular configuration, denote by $\mathbb{E}_i$, as the configuration that all distances between any two of p_i, p_{n1} and p_{n2} of $\mathbb{T}_i$ are equal to d_u.

3.2 Problem definition

It is known that the local geometric shape of schools of tuna represents a diamond shape (Stocker, 1999), whereby tuna exhibit their adaptive behavior while maintaining the local shape. Similarly, the local interaction in this work is to generate $\mathbb{E}_i$ from $\mathbb{T}_i$. Formally, the *local interaction* is to have r_i maintain d_u with N_i at each time toward forming $\mathbb{E}_i$. Now, we can

address the coordination problem of *adaptive navigation* of robot swarms based on the local interaction as follows:

Given $r_1,\cdots,r_n$ located at arbitrarily distinct positions, how to enable the robots to autonomously travel through unknown territories using only local information in order to reach a destination?

It is assumed that the unknown environment to be navigated by a swarm of robots includes obstacles and dead-end passageways. Next, we advocate that adaptive flocking can be achieved by solving the following four constituent sub-problems.

- **Problem-1**(*Maintenance*): Given robots located at arbitrarily distinct positions, how to enable them to navigate with $\mathbb{E}_i$.
- **Problem-2**(*Partition*): Given that an obstacle is detected, how to enable a swarm to split into multiple smaller swarms to avoid the obstacle.
- **Problem-3**(*Unification*): Given that multiple swarms exist in close proximity, how to enable them to merge into a single swarm.
- **Problem-4**(*Escape*): Given some robots trapped in a dead-end passageway, how to enable them to escape from the area.

4. Geometric local interaction scheme

This section explains the local interaction among three neighboring robots. As presented in ALGORITHM-1, the algorithm consists of a function $\varphi_{interaction}$ whose arguments are p_i and N_i at each activation.

constant d_u := uniform distance
FUNCTION $\varphi_{interaction}(\{p_{n1}, p_{n2}\}, p_i)$
1 $(p_{ct,x}, p_{ct,y}) := centroid(p_{n1}, p_{n2}, p_i)$
2 ϕ := angle between $\overline{p_{n1}p_{n2}}$ and r_i's local horizontal axis
3 $p_{ti,x} := p_{ct,x} + d_u \cos(\phi + \pi/2)/\sqrt{3}$
4 $p_{ti,y} := p_{ct,y} + d_u \sin(\phi + \pi/2)/\sqrt{3}$
5 $p_{ti} := (p_{ti,x}, p_{ti,y})$

Algorithm 1. Local Interaction (code executed by the robot r_i at point p_i)

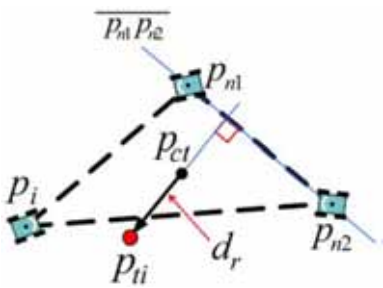


Fig. 3. Illustration of the local interaction algorithm

4.1 Local interaction algorithm

Let's consider a robot r_i and its two neighbors r_{n1} and r_{n2} located within r_i's SB. As shown in Fig. 3, three robots are configured into $\mathbb{T}_i$ whose vertices are p_i, p_{n1} and p_{n2}, respectively.

First, r_i finds the centroid of the triangle $\Delta p_i p_{n1} p_{n2}$, denoted by p_{ct}, with respect to its local coordinate system, and measures the angle ϕ between the line $\overline{p_{n1}p_{n2}}$ connecting the neighbors and $\vec{r}_{x,i}$ (r_i's horizontal axis). Using p_{ct} and ϕ, r_i calculates the next movement point p_{ti}. Each robot computes p_{ti} by its 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 running ALGORITHM-1, three robots configure themselves into $\mathbb{E}_i$(Lee & Chong, 2009).

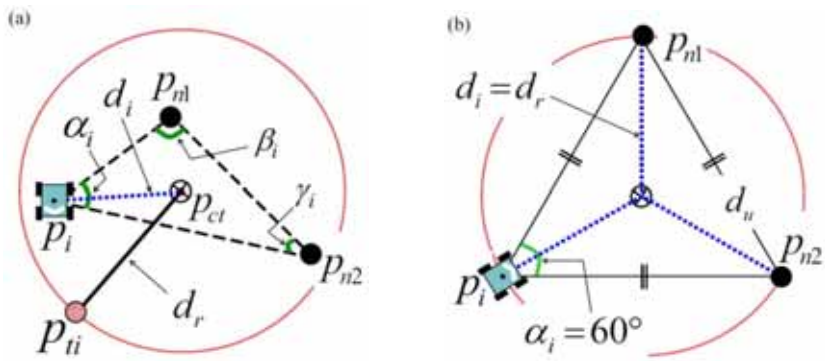
4.2 Motion control

As illustrated in Fig. 4-(b), let's consider the circumscribed circle of an equilateral triangle whose center is p_{ct} of $\Delta p_i p_{n1} p_{n2}$ and radius d_r is $d_u/\sqrt{3}$. Under the local interaction, the positions of each robot are determined by controlling the distance d_i from p_{ct} and the internal angle α_i(see Fig. 4-(a)). First, the distance is controlled by the following equation

$$\dot{d}_i(t)=-a(d_i(t)-d_r), \tag{1}$$

Where a is a positive constant. Indeed, the solution of (1) is $d_i(t)=|d_i(0)|e^{-at}+d_r$ that converges exponentially to d_r as t approaches infinity. Secondly, the internal angle is controlled by the following equation

$$\dot{\alpha}_i(t)=k\big(\beta_i(t)-\gamma_i(t)-2\alpha_i(t)\big), \tag{2}$$



(a) control parameters range d_i and bearing α_i (b) desired equilateral triangle $\mathbb{E}_i$

Fig. 4. Two control parameter of local interaction

where k is a positive number. Because the total internal angle of a triangle is 180°, (2) can be rewritten as

$$\dot{\alpha}_i(t)=k'\big(60°-\alpha_i(t)\big), \tag{3}$$

where k' is 3k. Likewise, the solution of (3) is $\alpha_i(t) = |\alpha_i(0)|e^{-k't} + 60°$ that converges exponentially to 60° as t approaches infinity.

Note that (1) and (3) imply that the trajectory of r_i converges to d_r and 60°, an equilibrium state as termed $[d_r \quad 60°]^T$ shown in Fig. 4-(b). This also implies that three robots eventually

form $\mathbb{E}_i$. In order to prove the convergence of the local interaction, we apply Lyapunov's stability theory (Slotine & Li, 1991). Lyapunov's stability theorem states that if there exists a scalar function $v(\mathbf{x})$ of the state $\mathbf{x}$ with continuous first order derivatives such that $v(\mathbf{x})$ is positive definite, $v'(\mathbf{x})$ is negative definite, and $v(\mathbf{x})\to\infty$ as $\|\mathbf{x}\|\to\infty$, then the equilibrium at the origin is asymptotically stable. Now, the desired configuration can be regarded as one that minimizes the energy level of a Lyapunov function.
Consider the following scalar function of the state $\mathbf{x} = [d_i(t) \quad \alpha_i(t)]^T$ with continuous first order derivatives:

$$f_{l,i} = \frac{1}{2}(d_i - d_r)^2 + \frac{1}{2}(60^\circ - \alpha_i)^2. \tag{4}$$

This scalar function is always positive definite except $d_i \neq d_r$ and $\alpha_i \neq 60$. The derivative of the scalar function is given by

$$\dot{f}_{l,i} = -(d_i - d_r)^2 - (60^\circ - \alpha_i)^2, \tag{5}$$

which is obtained by differentiating $f_{l,i}$ to substitute for $\dot{d}_i$ and $\dot{\alpha}_i$. It is evident that (5) is negative definite and the scalar function $f_{l,i}$ is radially unbounded since it tends to infinity as $\|\mathbf{x}\|\to\infty$. Therefore, the equilibrium state is asymptotically stable, implying that r_i reaches a vertex of $\mathbb{E}_i$. Further details on the convergence proof are given in (Lee & Chong, 2009).

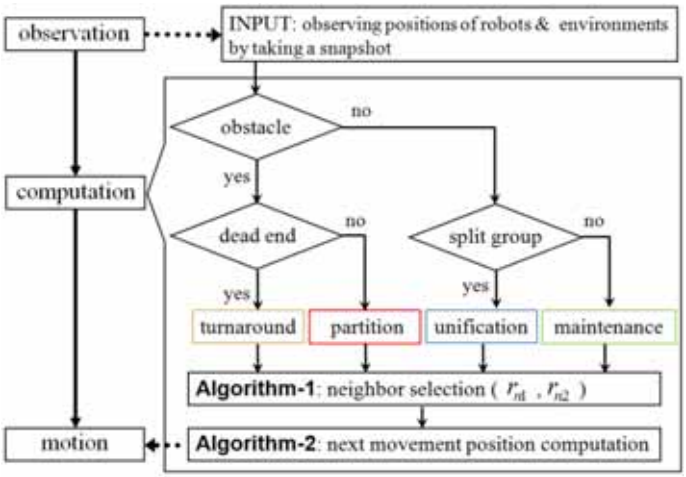


Fig. 5. Adaptive navigation algorithm flowchart

5. Adaptive navigation algorithm

As illustrated in Fig. 5, the input to the adaptive navigation algorithm at each time is 0_i and the arena border detected with respect to r_i's local coordinate system, and the output is r_i's next movement position. When r_i observes the arena within its SB, depending on whether or not it can move forward, it either executes the partition function to avoid the obstacle or the escape function to break a stalemate. When r_i faces no arena border but observes other

swarms, it executes the unification function. Otherwise, it basically performs the maintenance function to navigate toward a goal.

Function $\varphi_{rrsel}(O_i, p_i)$

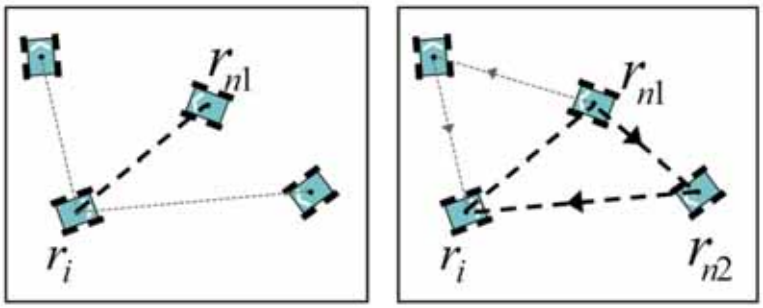
1 $p_{n1} := \min_{p \in O_i - \{p_i\}} [dist(p_i, p)]$

2 $p_{n2} := \min_{p \in O_i - \{p_i, p_{n1}\}} [dist(p_{n1}, p) + dist(p, p_i)]$

3 **Output:** $\{p_{n1}, p_{n2}\}$ $(= N_i)$

Algorithm 2. Neighbor Selection -1 (code executed by the robot r_i at point p_i)

The four functions above should determine two positions p_{n1} and p_{n2} occupied by two neighbors r_{n1} and r_{n2}. These positions are the input arguments of ALGORITHM-1. Before explaining the four functions as individual solutions of each sub-problem, we introduce the neighbor selection algorithm commonly used in the four functions, enabling r_i to select its neighbor robots. To form $\mathbb{T}_i$, the first neighbor r_{n1} is selected as the one located the shortest distance away from r_i as shown in Fig. 6-(a). When there exist two or more candidates r_{n1_m} and r_{n1_n} for r_{n1}, r_i arranges their positions $p_{n1_m}=(x_{n1_m},y_{n1_m})$ and $p_{n1_n}=(x_{n1_n},y_{n1_n})$ with respect to r_i's local coordinate system. Then, r_i uniquely determines its r_{s1} by sorting their positions in the following increasing order with respect to its local coordinate system: if $(p_{n1_m}<p_{n1_n}) \Leftrightarrow [(y_{n1_m}<y_{n1_n}) \vee \{(y_{n1_m}=y_{n1_n}) \wedge (x_{n1_m}<x_{n1_n})\}]$ where the logic symbols $\vee$ and $\wedge$ indicate the logical conjunction and the logical disjunction, respectively. As presented in Fig. 6-(b), the second neighbor r_{n2} is selected such that the length of $\mathbb{T}_i$'s perimeter is minimized as follows: $\min[dist(p_{n1},p_{n2})+dist(p_{n2},p_i)]$. In particular, when both r_i and the neighbors are all aligned, if there are three or more robots in O_i, r_{n2} is re-selected such that r_i is not located on the same line. Under ALGORITHM-2, r_i is able to select its neighbors and then form $\mathbb{T}_i$. Notice that the currently selected neighbors do not coincide with ones at the next time due to the assumption of anonymity. Using the current O_i by r_i, $\mathbb{T}_i$ is newly formed at each time.



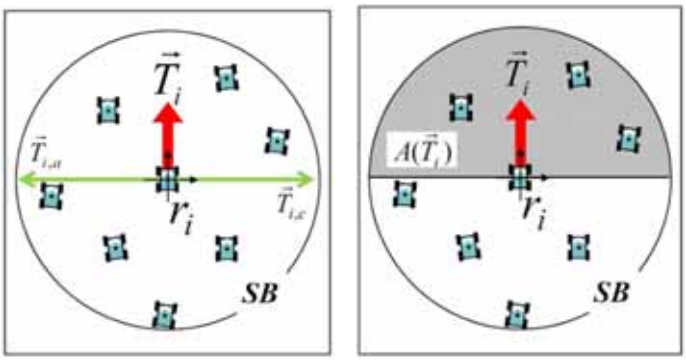
(a) r_{n1} selection (b) r_{n2} selection

Fig. 6. Illustration of the neighbour selection algorithm

5.1 Maintenance function

The first problem is how to maintain $\mathbb{E}_i$ with neighboring robots while navigating. As shown in Fig. 7-(a), r_i adjusts its traveling direction $\vec{T}_i$ with respect to its local coordinate system and

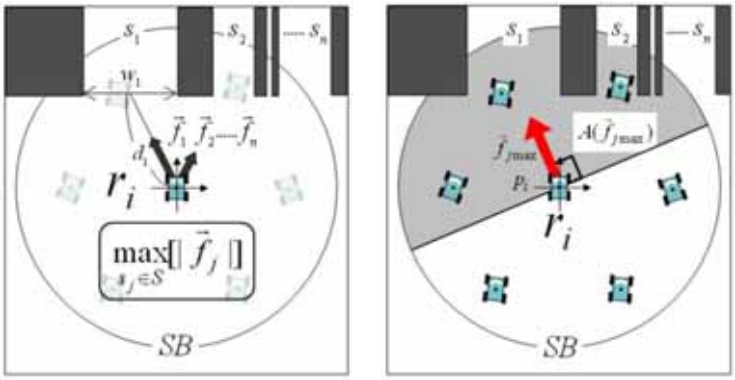
computes O_i at the time t. By rotating $\vec{T}_i$ 90 degrees clockwise and counterclockwise, respectively, two vectors $\vec{T}_{i,c}$ and $\vec{T}_{i,a}$ are defined. Within r_i's SB, an area of traveling direction $A(\vec{T}_i)$ is defined as the area between $\vec{T}_{i,c}$ and $\vec{T}_{i,a}$ as illustrated in Fig. 7-(b). Under ALGORITHM-2, r_i checks whether there exists a neighbor in $A(\vec{T}_i)$. If any robots exist within $A(\vec{T}_i)$, r_i selects the first neighbor r_{n1} and defines its position p_{n1}. Otherwise, r_i spots a virtual point p_v located some distance d_v away from p_i along $A(\vec{T}_i)$, which gives p_{n1}. After determining p_{n1}, r_{n2} is selected and its position p_{n2} is defined.



(a) traveling direction $\vec{T}_i$ (b) maintenance area $A(\vec{T}_i)$

Fig. 7. Illustration of the maintenance function

5.2 Partition function



(a) favorite vector $\vec{f}_j$ (b) partition area $A(\vec{f}_{j\max})$

Fig. 8. Illustration of the partition function

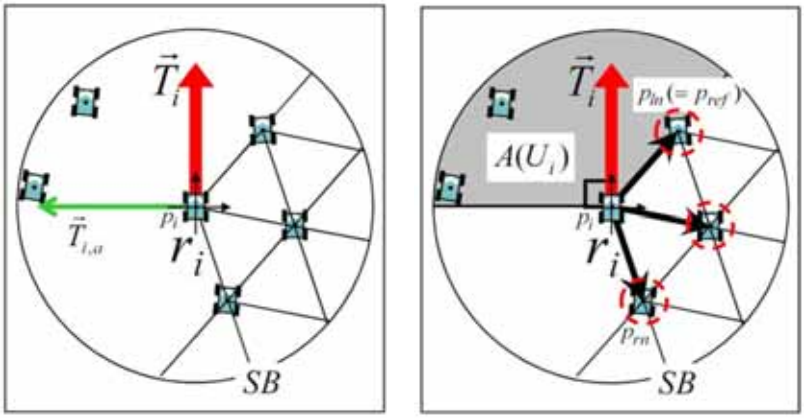
When r_i detects an obstacle that blocks its way to the destination, it is required to modify the direction toward the destination avoiding the obstacle. In this work, r_i determines its direction by using the relative degree of attraction of individual passageways s_j, termed the favorite vector $\vec{f}_j$, whose magnitude is given:

$$|\vec{f}_j| = \left|\frac{w_j}{d_j^2}\right| \tag{6}$$

where w_j and d_j denote the width of s_j and the distance between the center of w_j and p_i, respectively. Note that if r_i can not exactly measure w_j beyond its SB, w_j may be shortened. Now, s_j can be represented by a set of favorite vectors $\{\vec{f}_j | 1 \le j \le m\}$, and then r_i selects the maximum magnitude of $\vec{f}_j$, denoted as $|\vec{f}_j|_{max}$. Similar to defining $A(\vec{T}_i)$ above, r_i defines a maximum favorite area $A(\vec{f}_{jmax})$ based on the direction of $|\vec{f}_j|_{max}$ within its SB. If neighbors are found in $A(\vec{f}_{jmax})$, r_i selects r_{n1} to define p_{n1}. Otherwise, r_i spots a virtual point p_v located at d_v in the direction of $|\vec{f}_j|_{max}$ to define p_{n1}. Finally, r_{n2} and its r_{n2} are determined under ALGORITHM-2.

5.3 Unification function

In order to enable multiple swarms in close proximity to merge into a single swarm, r_i adjusts $\vec{T}_i$ with respect to its local coordinate system and defines the position set of robots D_u located within the range of d_u. r_i computes $ang(\vec{T}_i, \overrightarrow{p_i p_u})$, where $\overrightarrow{p_i p_u}$ is the vector starting from p_i to a neighboring point p_u in D_u, and defines a neighbor point p_{ref} that gives the minimum $ang(\vec{T}_i, \overrightarrow{p_i p_u})$ between $\vec{T}_i$ and $\overrightarrow{p_i p_u}$. If there exists p_{ul}, r_i finds another neighbor point p_{um} using the same method starting from $\overrightarrow{p_i p_{ul}}$. Unless p_{ul} exists, r_i defines p_{ref} as p_{rn}. Similarly, r_i can decide a specific neighbor point p_{ln} while rotating 60 degrees counterclockwise from $\overrightarrow{p_i p_{ref}}$. The two points, denoted as p_{rn} and p_{ln}, are located at the farthest point in the right-hand or left-hand direction of $\overrightarrow{p_i p_u}$, respectively. Next, a unification area $A(U_i)$ is defined as the common area between $A(\vec{T}_i)$ 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 $A(U_i)$ and selects the first neighbor r_{n1}. In particular, the second neighbor position p_{n2} is defined such that the total distance from p_{n1} to p_i can be minimized only through either p_{rn} or p_{ln}.

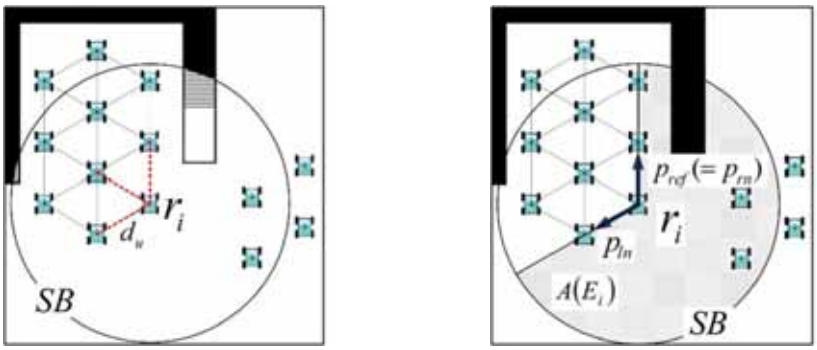


(a) traveling direction $\vec{T}_i$ (b) unification area $A(U_i)$

Fig. 9. Illustration of the unification function

5.4 Escape control

When r_i detects an arena border within its SB as shown in Fig. 10-(a), it checks whether $\mathbb{T}_i$ is equal to $\mathbb{E}_i$. Neighboring robots should always be kept d_u distance from r_i. Moreover, r_i's current position p_i and its next movement position p_{ti} remain unchanged for several time steps, r_i will find itself trapped in a dead-end passageway. r_i then attempts to find new neighbors within the area $A(E_i)$ to break the stalemate. Similar to the unification function, r_i adjusts $\vec{T}_i$ with respect to its local coordinate system and defines the position set of robots D_e located within SB. As shown in Fig. 10-(b), r_i computes $\mathrm{ang}(\vec{T}_i, \overrightarrow{p_i p_e})$, where $\overrightarrow{p_i p_u}$ is the vector starting from p_i to a neighboring point p_e in D_e, and defines a neighbor point r_{ref} that gives the minimum $\mathrm{ang}(\vec{T}_i, \overrightarrow{p_i p_e})$ between $\vec{T}_i$ and $\overrightarrow{p_i p_u}$. While rotating 60 degrees clockwise and counterclockwise from $\overrightarrow{p_i p_{ref}}$, respectively, r_i can decide the specific neighbor points p_{ln} and p_{rn}. Employing p_{ln} and p_{rn}, the escape area $A(E_i)$ is defined. Then, r_i adjusts a set of robots in $A(E_i)$ and selects the first neighbor r_{n1}. In particular, the second neighbor position p_{n2} is determined under ALGORITHM-2.



(a) encountered dead-end passageway (b) merging with another adjacent swarm

Fig. 10. Illustration of the escape function

6. Simulation results and discussion

This section describes simulation results that tested the validity of our proposed adaptive navigation scheme. We consider that a swarm of robots attempts to navigate toward a stationary goal while exploring and adapting to unknown environmental conditions. In such an application scenario, the goal is assumed to be either a light or odor source that can only be detected by a limited number of robots. As mentioned in Section 3, the coordinated navigation is achieved without using any leader, identifiers, global coordinate system, and explicit communication. We set the range of SB to 2.5 times longer than d_u.

The first simulation demonstrates how a swarm of robots adaptively navigates in an environment populated with obstacles and dead-end passageway. In Fig. 11, the swarm navigates toward the goal located on the right hand side. On the way to the goal, some of the robots detect a triangular obstacle that forces the swarm split into two groups from 7 sec (Fig. 11-(c)). The rest of the robots that could not identify the obstacle just follow their neighbors moving ahead. After being split into two groups at 14 sec (Fig. 11-(d)), each group maintains their local geometric configuration while navigating. At 18 sec (Fig. 11-(e)), some

robots happen to enter a dead-end passageway. After they find themselves trapped, they attempt to escape from the passageway by just merging themselves into a neighboring group from 22 sec to 32 sec (from Figs. 11-(f)) to (k)). After 32 sec (Fig. 11-(k)), simulation result shows that two groups merge again completely. At 38 sec (Fig. 11-(l)), the robots successfully pass through the obstacles.

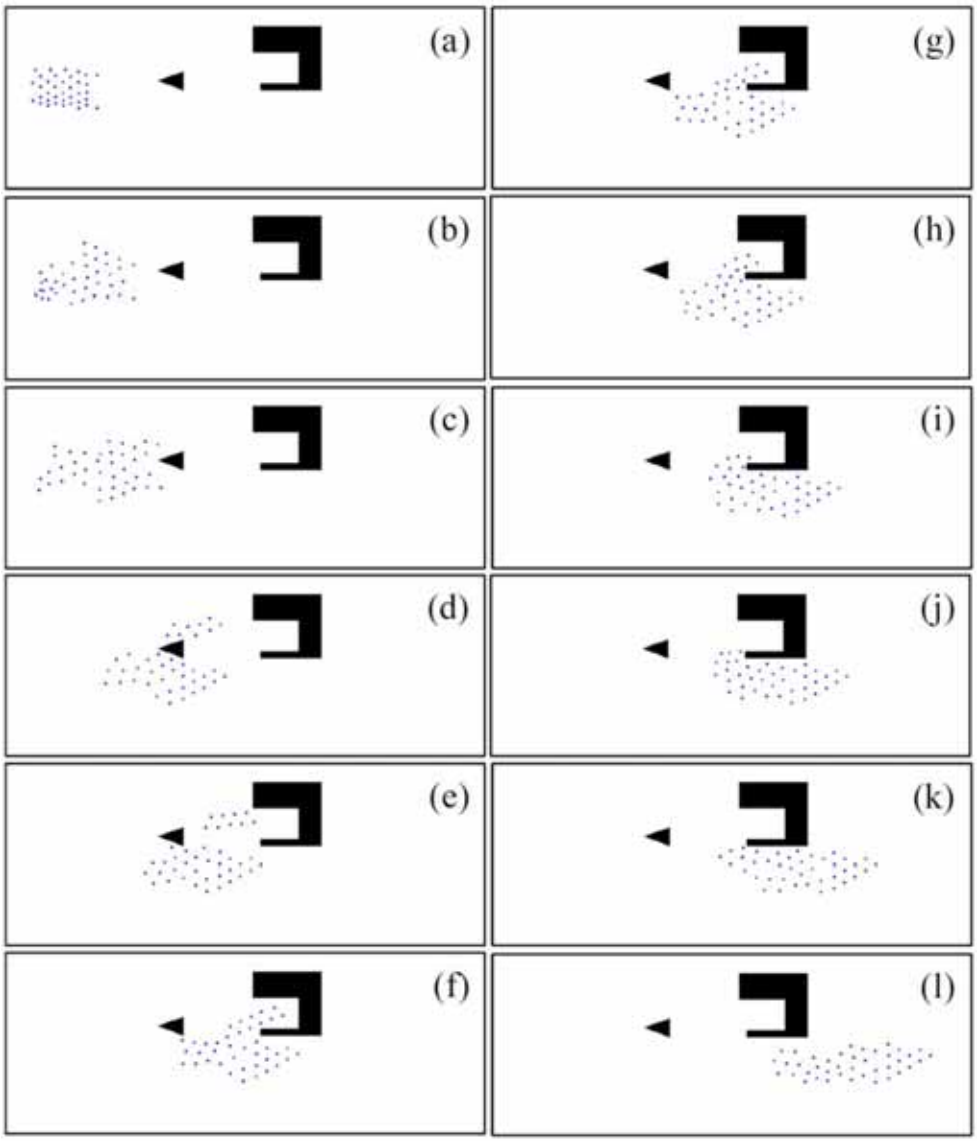


Fig. 11. Simulation results of adaptive flocking toward a stationary goal ((a)0 sec,(b)4 sec, (c)7 sec,(d)14 sec,(e)18 sec,(f)22 sec,(g)23 sec,(h)24 sec,(i)28 sec,(j)29 sec,(k)32 sec,(l)38sec)

Fig. 12 shows the trajectories of individual robots in Fig. 11. We could confirm that the swarm was split into two groups due to the triangular obstacle located at coordinates (*0,0*). If we take a close look at Figs. 11-(f) through (j) (from 22 sec to 29 sec), the trapped ones escaped from the dead-end passageway located at coordinates (x, 200). More important, after passing through the obstacles, all robots position themselves from each other at the desired interval d_u.

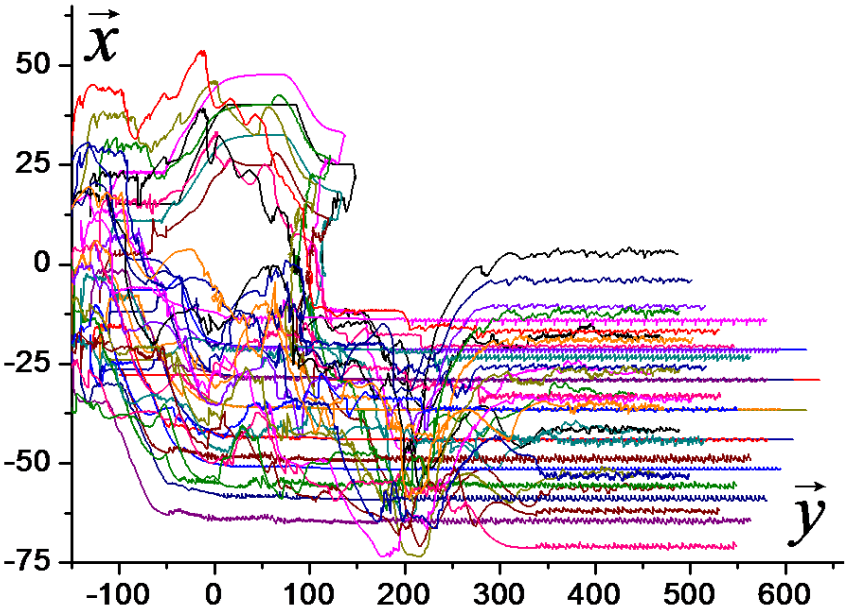


Fig. 12. Robot trajectory results for the simulation in Fig.11

Next, the proposed adaptive navigation is evaluated in a more complicated environmental condition as presented in Fig. 13. On the way to the goal, some of the robots detect a rectangular obstacle that forces the swarm split into two groups in Fig. 13-(b). After passing through the obstacle in Fig. 13-(d), the lower group encounters another obstacle in Fig. 13-(e), and split again into two smaller groups in Fig. 13-(g). Although several robots are trapped in a dead-end passageway, their local motions can enable them to escape from the dead-end passageway in Fig. 13-(i). This self-escape capability is expected to be usefully exploited for autonomous search and exploration tasks in disaster areas where robots have to remain connected to their *ad hoc* network. Finally, for a comparison of the adaptive navigation characteristics, three kinds of simulations are performed as shown in Figs. 14 through 16. All the simulation conditions are kept the same such as *du*, the number of robots, and initial distribution. Fig. 14 shows the behavior of mobile robot swarms without the partition and escape functions. Here, a considerable number of robots are trapped in the dead-end passageway and other robots pass through an opening between the obstacle and the passageway by chance. As compared with Fig. 14, Fig. 15 shows more robots pass through the obstacles using the partition function. However, a certain number of robots remain trapped in the dead-end passageway because they have no self-escape function. Fig.

16 shows that all robots successfully pass through the obstacles using the proposed adaptive navigation scheme. It is evident that the partition and escape functions will provide swarms of robots with a simple yet efficient navigation method. In particular, self-escape is one of the most essential capabilities to complete tasks in obstacle-cluttered environments that require a sufficient number of simple robots.

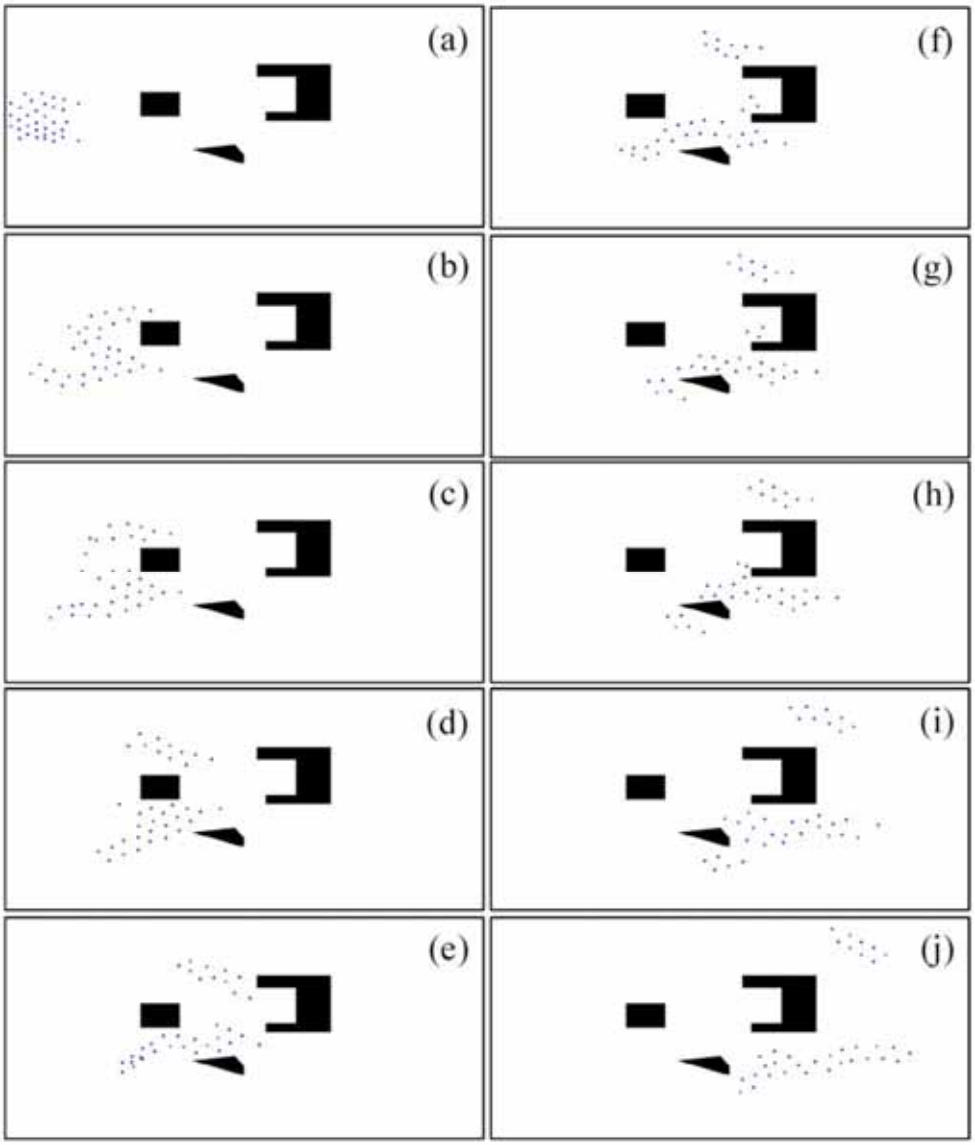


Fig. 13. Simulation results of adaptive flocking toward a stationary goal ((a)0 sec,(b)8 sec, (c)10 sec,(d)14 sec,(e)18 sec,(f)22 sec,(g)25 sec,(h)27 sec,(i)31 sec,(j)36)

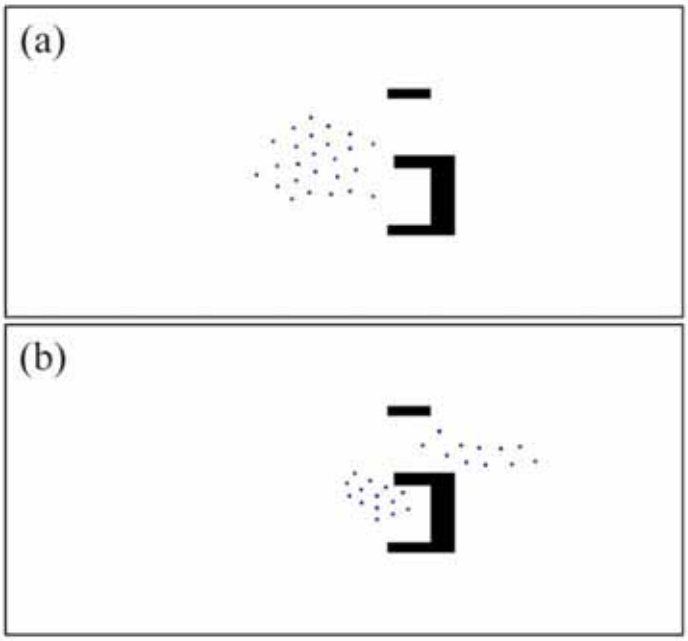


Fig. 14. Simulation results for flocking without partition and escape functions

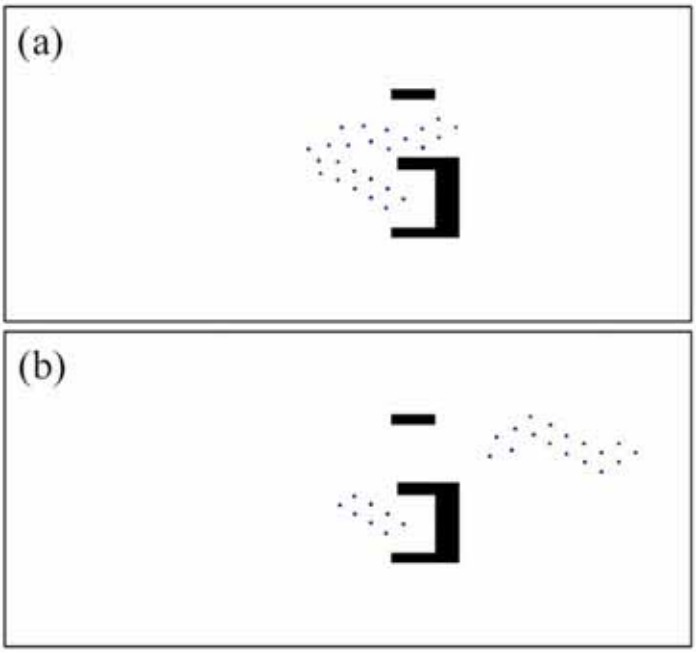


Fig. 15. Simulation results for flocking with only partition function

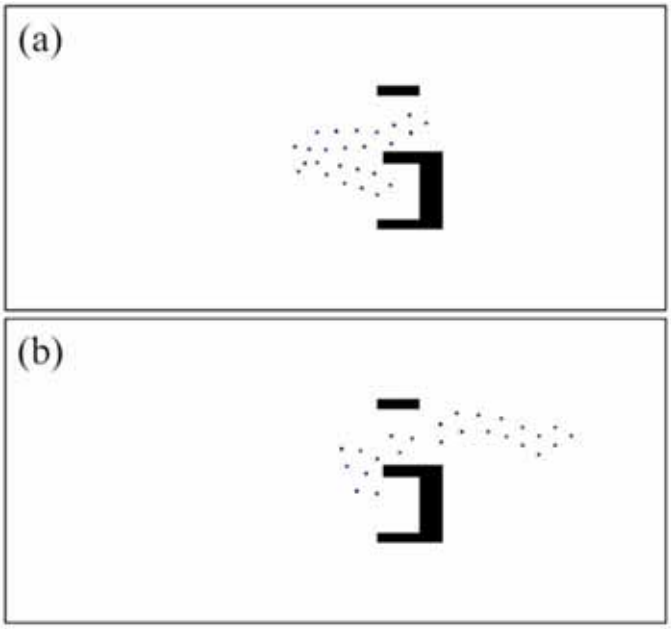


Fig. 16. Simulation results for flocking with the partition and escape functions

7. Conclusions

This paper was devoted to developing a new and general coordinated adaptive navigation scheme for large-scale mobile robot swarms adapting to geographically constrained environments. Our distributed solution approach was built on the following assumptions: anonymity, disagreement on common coordinate systems, no pre-selected leader, and no direct communication. The proposed adaptive navigation was largely composed of four functions, commonly relying on dynamic neighbor selection and local interaction. When each robot found itself what situation it was in, individual appropriate ranges for neighbor selection were defined within its limited sensing boundary and the robots properly selected their neighbors in the limited range. Through local interactions with the neighbors, each robot could maintain a uniform distance to its neighbors, and adapt their direction of heading and geometric shape. More specifically, under the proposed adaptive navigation, a group of robots could be trapped in a dead-end passage, but they merge with an adjacent group to emergently escape from the dead-end passage. Furthermore, we verified the effectiveness of the proposed strategy using our in-house simulator. The simulation results clearly demonstrated that the proposed algorithm is a simple yet robust approach to autonomous navigation of robot swarms in highly-cluttered environments. Since our algorithm is local and completely scalable to any size, it is easily implementable on a wide variety of resource-constrained mobile robots and platforms. Our adaptive navigation control for mobile robot swarms is expected to be used in many applications ranging from examination and assessment of hazardous environments to domestic applications.

8. References

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

Correll, N., Bachrach, J., Vickery, D., & Rus, D. (2009). Ad-hoc wireless network coverage with networked robots that cannot localize, *Proc. IEEE Int. Conf. Robotics and Automation*, pp. 3878 - 3885, IEEE

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

Folino, G. & Spezzano, G. (2002). An adaptive flocking algorithm for spatial clustering, In: *Parallel Problem Solving from Nature* (LNCS), Yao, X., Burke, E., Lozano, J. A., Smith, J., Merelo-Guervos, J. J., Bullinaria, J. A., Rowe, J., Tino, P., Kaban, A., & Schwefel, H.-P. (Ed.), Vol. 2439, 924-933, Springer Berlin, ISBN: 978-3-540-23092-2

Fowler, T. (2001). Mesh networks for broadband access, IEE Review, Vol. 47, No. 1, 17-22

Gu, D. & Hu, H. (2010). Distributed minmax filter for tracking and flocking, *Proc. IEEE/RSJ Int. Conf. Intelligent Robots and Systems*, pp. 3562-3567, IEEE

Ikemoto, Y., Hasegawa, Y., Fukuda, T., & Matsuda, K. (2005). Graduated spatial pattern formation of robot group, *Information Science*, Vol. 171, No. 4, 431-445

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

Lee, G. & Chong, N. Y. (2009). A geometric approach to deploying robot swarms, *Annals of Mathematics and Artificial Intelligence*, Vol. 52, No. 2 − 4, 257-280

Lee, G. & Chong, N. Y. (2009-b). Decentralized formation control for small-scale robot teams with anonymity, *Mechatronics*, Vol. 19, No. 1, 85-105

Lee, G. & Chong, N. Y. (2008). Adaptive flocking of robot swarms: algorithms and properties, *IEICE Trans. Communications*, Vol. E91 − B, No. 9, 2848-2855

Lyengar, R., Kar, K., & Banerjee, S. (2005). Low-coordination topologies for redundancy in sensor networks, *Proc. 3rd Acm Int. Sym. Mobile Ad Hoc Networking and Computing*, pp. 332-342, ACM

Nembrini, J., Winfield, A., & Melhuish, C. (2002). Minimalist coherent swarming of wireless networked autonomous mobile robots, *Proc. 7th Int. Conf. Simulation of Adaptive Behavior*, pp. 373-382, IEEE

Olfati-Saber, R. (2006). Flocking for mult-agnet dynamic systems: algorithms and theory, *IEEE Trans. Automatic Control*, Vol. 51, No. 3, 401-420

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

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

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

Shucker, B., Murphey, T. D., & Bennett, J. K. (2008). Convergence-preserving switching for topology-dependent decentralized systems, *IEEE Trans. Robotics*, Vol. 24, No. 6, 1405-1415

Slotine, J. E. & Li,W. (1991). Applied nonlinear control, Prentice-Hall, ISBN: 0-13-040890-5

Spears, D., Kerr, W., & Spears, W. M. (2006). Physics-based robot swarms for coverage problems, *Int. Jour.Intelligent Control and Systems*, Vol. 11, No. 3, 124-140

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

Stocker, S. (1999). Models for tuna school formation, *Mathematical Biosciences*, Vol. 156, No. 1−2, 167-190

Wilson, E. O. (1976). Sociobiology: the new synthesis, Harvard University Press, ISBN: 0-674-00089-7

Yicka, J., Mukherjeea, B., & Ghosal, D. (2008). Wireless sensor network survey, *Computer Networks*, Vol. 52, No. 12, 2292-2330

Zarzhitsky, D., Spears, D., & Spears, W. M. (2005). Distributed robotics approach to chemical plume tracing, *Proc. IEEE/RSJ Int. Conf. Intelligent Robots and Systems*, pp. 4034-4039, IEEE