Self-Configurable Mobile Robot Swarms: Adaptive Triangular Mesh Generation

Geunho Lee and Nak Young Chong

School of Information Science

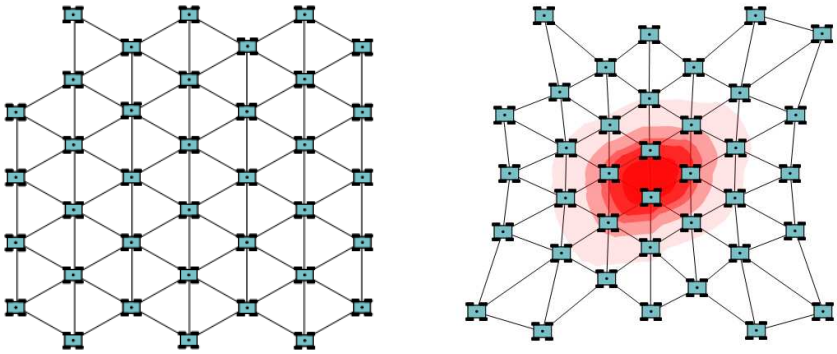
Japan Advanced Institute of Science and Technology

Ishikawa, Japan

Geunho Lee and Nak Young Chong (2013). Self-Configurable Mobile Robot Swarms: Adaptive Triangular Mesh Generation. In Nak Young Chong (Eds.) Networking Humans, Robots and Environments. Bentham Science Publishers, pp.59-75. http://dx.doi.org/10.2174/97816080573131130101

ABSTRACT. We address the problem of dispersing a large number of autonomous mobile robots, for building wireless *ad hoc* sensor networks performing environmental monitoring and control. For this purpose, we propose the adaptive triangular mesh generation algorithm that enables robots to generate triangular meshes of various sizes, adapting to changing environmental conditions. A locally interacting, geometric technique allows each robot to generate a triangular mesh with its two neighbor robots. Specifically, we have assumed that robots are not allowed to have any identifiers, any pre-determined leaders or common coordinate systems, or any explicit communication. Under such minimal conditions, the positions of the robots were shown to converge to the desired distribution. This convergence was mathematically proven and also verified through extensive simulations. Our results indicate that the proposed algorithm can be applied to problems regarding the coverage of an area of interest by a swarm of mobile sensors.

1. Introduction



(a) uniform deployment (b) adaptive deployment

FIGURE 1. Uniform vs. adaptive triangular meshes of mobile robot swarms

With the advance of wireless and mobile networking technologies, much attention has been paid to the use of large-scale swarms of simple, low-cost mobile robots for many applications [**1**] [**2**]. With the goal of deployment of robot swarms in real environments, researchers in swarm robotics have recently presented many fundamental coordination approaches such as self-configuration [**9**] [**16**], pattern formation [**18**] [**19**], flocking [**20**] [**21**], and consensus [**22**] [**23**]. Based on these coordination approaches, robot swarms are expected to perform a wide variety of real tasks, such as environmental or habitat monitoring [**14**] [**17**], exploration [**24**], search-and-rescue [**25**], odor localization [**26**]- [**28**], and so on.

In particular, for environmental or habitat monitoring, self-configuration of robot swarms requires a type of collective behavior that allows robots to disperse themselves in a certain area at a uniform spatial density. Thus, it is essential to properly coordinate the (relative) positions of robots, and this issue has been widely reported in literature [**5**]- [**15**]. Taking steps to further improve on those previous approaches, this work is aimed at presenting an algorithm to enable robot swarms to configure themselves adaptively in an area of interest

with varying spatial densities. As illustrated in Fig. 1, robot swarms can explore an unknown area and detect and sense oil or chemical spills across the area. The contaminated area should be covered efficiently with mobile robots or sensors to investigate the degree and extent of contamination as quickly as possible and, if possible, prevent the possible expansion of the area. Therefore, in this paper, we address the problem of how to enable swarms of autonomous mobile robots to self-adjust their configuration or spatial density to fit local environmental conditions.

Based on our prior research on swarm configuration [**16**] [**17**], we propose an adaptive triangular mesh generation algorithm that enables robot swarms to explore an area, and to adjust the intervals between neighboring robots autonomously. The main objective is to provide robots with adaptive deployment capabilities to cover an area of interest more efficiently with variable triangular meshes according to sensed area conditions. This also can give us a more accurate picture of variations in the conditions of an area. In this chapter, the properties of the proposed algorithm are mathematically explained and the convergence is proven. We also demonstrated through extensive simulations that a large-scale swarm of robots can establish a triangular mesh network, adapting to varying degrees of connection. The results have been encouraging, and indicate that self-configurable robot swarms can be deployed for environmental or habitat monitoring.

The rest of this chapter is organized as follows. Section 2 gives a brief description on the state-of-the-art in the field of robot swarms. Section 3 presents the formal definitions of the adaptive triangular mesh generation problem. Section 4 describes our approach, its mathematical properties, and convergence at the equilibrium state. Section 5 summarizes the results of simulations. Section 6 explains our conclusions.

2. Background

Decentralized control for robot swarms can be broadly classified into global and local strategies, according to whether sensors have range limits. Global strategies [**3**] [**4**] [**19**] may provide fast, accurate, and efficient deployment, but are technically infeasible, and lack scalability as the number of robots increases. On the other hand, local strategies are mainly based on interactions between individual robots, inspired by nature. Local strategies can further be divided into biological emergence [**5**] [**6**], behavior-based [**7**], and virtual physics-based [**8**]- [**15**] approaches. Many of the behavior-based and virtual physics-based approaches use such physical phenomena as electric charges [**8**], gravitational forces [**9**], spring forces [**10**] [**14**] [**15**], potential fields [**11**], van der Waals forces [**12**], and other virtual models [**13**].

Robot swarm configurations achieved by the above-mentioned local interactions may result in lattice-type networks. These configurations offer high level coverage and multiple redundant connections, ensuring maximum reliability and flexibility from the standpoint of topology. Depending on whether there are interactions among all robots, the networks can be classified as fully or partially-connected topologies [**30**]. The fully-connected topologies have each robot interact with all other robots within a certain range simultaneously. Thus, those approaches might over-constrain individual robots, and frequently lead to deadlocks. However, using the partially-connected topology, robots interact selectively with other robots, but are connected to all robots. For example, robots may choose to exert force in a certain direction [**14**], where this selective interaction helps prevent them from being too tightly constrained. Therefore, robots may be able to achieve faster formation without deadlocks [**15**].

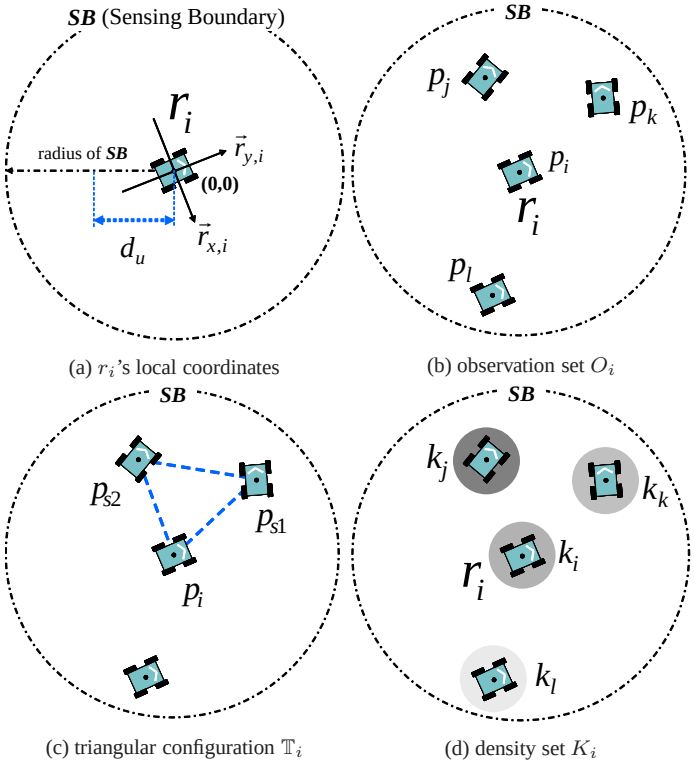
In our earlier work [**17**], we presented self-configuration of a robot swarm which enables a large number of robots to configure themselves into a 2-dimensional plane with geographic constraints. A locally-interacting geometric technique based on partially-connected topology provides a unique solution that allows robots to converge to uniform distribution by forming an equilateral triangle with two neighbors. By collecting this local behavior of each robot, a uniformly spaced swarm of robots was organized to cover an environment. Unlike previous works [**5**]- [**15**], our approach first was to construct uniformly spaced equilateral triangles conforming to the borders of an unknown area, when the robot sensors are subject to range and accuracy limitations. Second, an equilateral triangle lattice is built, with a partially-connected mesh topology. Among all the possible types of regular polygons, equilateral triangle lattices can reduce the computational burden, and are less influenced by other robots, due to the limited number of neighbors, and are highly scalable. The proposed local interaction is computationally efficient, since each robot utilizes only position information of two other robots. Our approach eliminates such major assumptions as robot identifiers, common coordinates, global orientation, and direct communication. More specifically, robots compute the target positions without requiring memories of past actions or states, helping to cope with transient errors.

3. Problem Statement

Definition and Notation. We consider a swarm of mobile robots, denoted as $r_1, \cdots, r_n$. It is assumed that all robots are within a swarm network configured by our previously proposed self-configuration method [**16**] [**17**]. Each robot autonomously moves on a 2-dimensional plane. They have no leader and no identifiers, do not share any common coordinate system, and do not retain any memory of past actions. Due to limited sensing range, they can detect the position of other robots only within a certain distance. In addition, robots do not communicate explicitly with other robots.

Let us consider a *robot* r_i with local coordinates $\vec{r}_{x,i}$ and $\vec{r}_{y,i}$, as illustrated in Fig. 2-(a). Here, $\vec{r}_{y,i}$ defines the vertical axis of r_i's coordinate system as its heading direction. It is straightforward to determine the horizontal axis $\vec{r}_{x,i}$ by rotating the vertical axis 90 degrees counterclockwise. The *position* of r_i is denoted as p_i. Note that p_i is $(0, 0)$ with respect to r_i's local coordinates. The *distance* between p_i of r_i and p_j of another robot r_j is denoted as $dist(p_i, p_j)$. We define a *uniform distance* d_u, the predefined desired interval between r_i and r_j. 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 an *observation set* of the positions O_i (=$\{p_j, p_k, p_l\}$) with respect to its local coordinates. Next, r_i can select two robots r_{s1} and r_{s2} within r_i's *SB*, which we call the *neighbors* of r_i, and denote their positions, $\{p_{s1}, p_{s2}\}$, as N_i. Given p_i and N_i in Fig. 2-(c), the *triangular configuration*, denoted by $\mathbb{T}_i$, is defined as a set of three distinct positions $\{p_i, p_{s1}, p_{s2}\}$.

As mentioned above, if robots detect an event such as an oil or chemical spill within the self-configured network, they attempt to cooperate with each other to cover the area as efficiently as possible. The gradient in contamination density across the area forces robots to adjust the intervals between neighboring robots. For a certain point p_i occupied by r_i as presented in Fig. 2-(d), the densities are expressed by k_i ranging between $0 < k_i \leq 1$, where $k_i = 0$ represents the maximum density and $k_i = 1$ corresponds to zero density. Moreover, it is assumed that each robot can detect and measure densities for positions occupied by other robots within *SB*, yielding a set of the densities K_i corresponding to individual positions in O_i.



(a) r_i's local coordinates (b) observation set O_i

(c) triangular configuration $\mathbb{T}_i$ (d) density set K_i

FIGURE 2. Illustration of definitions and notations frequently used in this chapter

Problem Definition. Now, we formally address the ADAPTIVE TRIANGULAR MESH GENERATION problem as follows.

Given a swarm of mobile robots self-configured in a 2-dimensional plane, how can the robots form triangular mesh patterns of various sizes adapting to varying environmental conditions?

Our approach to the above problem enables robots to disperse themselves into equilateral triangular patterns of various sizes, according to changes in environmental conditions within an area of interest. The basic concept behind this approach is to enable robots to form triangular configurations while changing their suitable distances depending on locally-sensed information. In other words, given a uniform distance d_u, three neighboring robots configure an equilateral triangle with a side length proportional to the local measurement data density.

4. Solution Approach

Regarding our solution approach, this section explains the generation of triangular meshes adapting to the measurement data densities in the areas occupied by three neighboring robots, provides two important properties of the approach, and shows the convergence into a desired stable configuration using Lyapunov's theory.

ALGORITHM-1 ADAPTIVE TRIANGULAR MESH GENERATION

Function $\varphi_{triangular}(O_i, K_i)$

$p_{s1} := \min_{p \in O_i - \{p_i\}} [dist(p_i, p)]$

$p_{s2} := \min_{p \in O_i - \{p_i, p_{s1}\}} [dist(p_{s1}, p) + dist(p, p_i)]$

$\phi :=$ angle between $\overline{p_{s1}p_{s2}}$ and r_i's local horizontal axis

$p_{ct} := (p_{ct,x}, p_{ct,y})$ // *centroid* of $\mathbb{T}_i$ (= $\{p_{s1}, p_{s2}, p_i\}$)

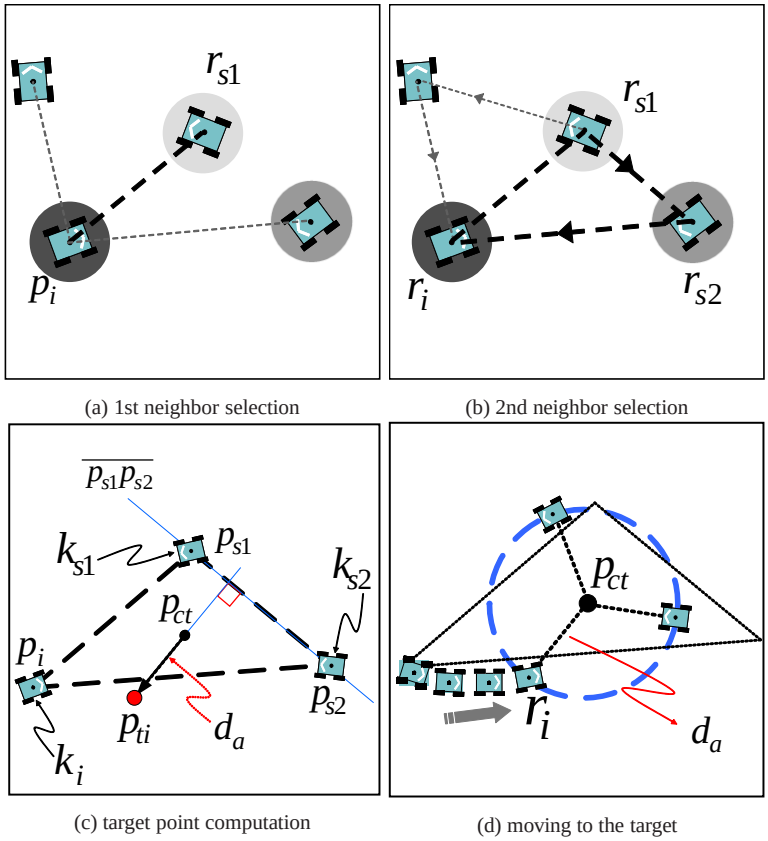
$k_a := \frac{k_i + k_{s1} + k_{s2}}{3}$ // average contamination density

$d_a := k_a \times \frac{d_u}{\sqrt{3}}$ // desired interval from p_{ct}

$p_{ti,x} := p_{ct,x} + d_a \cos(\phi + \frac{\pi}{2})$

$p_{ti,y} := p_{ct,y} + d_a \sin(\phi + \frac{\pi}{2})$

$p_{ti} := (p_{ti,x}, p_{ti,y})$ // next target point



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

(c) target point computation (d) moving to the target

FIGURE 3. Illustration of ALGORITHM-1

Algorithm Description. Here, we describe the adaptive triangular mesh generation algorithm. As presented in ALGORITHM-1, the algorithm consists of a function $\varphi_{triangular}$ whose arguments are O_i and K_i at each activation. Each time, r_i first observes other robots within O_i to select the closest neighbor r_{s1} as illustrated in Fig. 3-(a). If there exist two or more candidates for r_{s1}, r_i determines r_{s1} according to the high contamination density. Secondly, as illustrated in Fig. 3-(b), the second neighbor r_{s2} within O_i is selected, such

that the total distance from the position p_{s1} of r_{s1} to p_i passing through p_{s2} is minimized. Similarly, if there are two or more candidates for r_{s2}, r_i selects r_{s2} occupied in the area with higher density of contamination. Then, as illustrated in Fig. 3-(c), r_i measures the angle ϕ between the line $\overline{p_{s1}p_{s2}}$ connecting two neighbors and the horizontal axis of the observing r_i's coordinate system. Thirdly, the centroid p_{ct} in $\mathbb{T}_i$ ($\triangle p_i p_{s1} p_{s2}$) is computed. Moreover, based on k_i, k_{s1}, and k_{s2} at each position occupied by r_i, r_{s1}, and r_{s2}, r_i finds the local average density k_a in $\mathbb{T}_i$ through the computation of $(k_i + k_{s1} + k_{s2})/3$. Then, from p_{ct}, r_i calculates an appropriate interval as follows: $d_a = k_a \times d_u/\sqrt{3}$. Utilizing d_a and ϕ, r_i calculates its target point $p_{ti} = (p_{ti,x}, p_{ti,y})$ located on the same line as the previously calculated interval from p_{ct} and perpendicular to $\overline{p_{s1}p_{s2}}$. Finally, r_i moves toward p_{ti} as illustrated in Fig. 3-(d). By repeating this process, r_i can form a triangular mesh depending on the contamination densities.

Mathematical Properties. Let's consider a triangle (whose centroid is p_{ct}) $\triangle p_i p_{s1} p_{s2}$ (or $\mathbb{T}_i$) configured from the three positions occupied by r_i, r_{s1}, and r_{s2}. By ALGORITHM-1 above, at time t, r_i in $\mathbb{T}_i(t)$ finds the next target point p_{ti} of which the line segment $\overline{p_{ct}p_{ti}}$ is $k_a d_u/\sqrt{3}$ in length and is perpendicular to $\overline{p_{s1}p_{s2}}$ in Fig. 3-(c). In other words, at $t+1$, the height of $\triangle p_{ti} p_{s1} p_{s2}$ is the straight line through p_{ti} and perpendicular to $\overline{p_{s1}p_{s2}}$. Similarly, since r_{s1} and r_{s2} also execute the same algorithm, it is easily seen that p_{ct} at t is the orthocenter H at $t+1$.

In Fig. 4, we denote p_i, p_{s1}, and p_{s2} for simplicity as A, B, and C, respectively. The lengths of lines $\overline{AB}$, $\overline{AC}$, and $\overline{BC}$ are denoted as c, b, and a, respectively. The points P, Q, and R are the foot of the perpendiculars from the vertices C, B, and A to the vectors $\overrightarrow{AB}$, $\overrightarrow{AC}$, and $\overrightarrow{BC}$, respectively. The angles $\angle CAB$, $\angle ABC$, and $\angle BCA$ are denoted as α, β, and γ, respectively. Moreover, H is the orthocenter of $\triangle ABC$. Moreover, H is the orthocenter of $\triangle ABC$ (Since p_{ct} and H exist in the same location under ALGORITHM-1, we also use only H instead of p_{ct}).

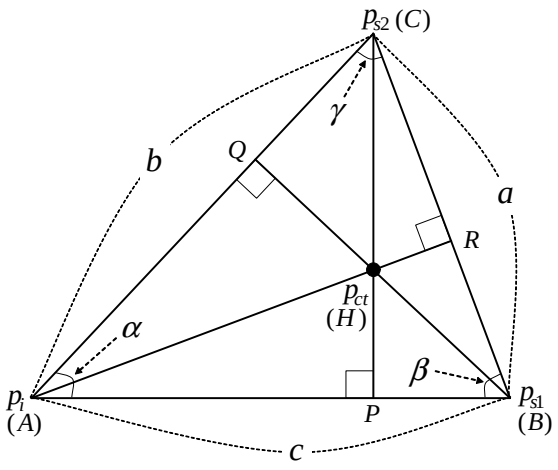


FIGURE 4. Illustrating the derivation of r_i's position vector toward H

Since $\overrightarrow{AB}$ and $\overrightarrow{AC}$ are linearly independent, $\overrightarrow{AH}$ can be defined as

$$\overrightarrow{AH} = x\overrightarrow{AB} + y\overrightarrow{AC}, \tag{1}$$

where *x* and *y* are scaling coefficients. Since we can easily see that $\overrightarrow{AP} = \frac{b\cos\alpha}{c}\overrightarrow{AB}$, the following relation holds:

$$\overrightarrow{PH} = \overrightarrow{AH} - \overrightarrow{AP} = (x - \tfrac{b\cos\alpha}{c})\overrightarrow{AB} + y\overrightarrow{AC}.$$

Thus, the inner product between $\overrightarrow{PH}$ and $\overrightarrow{AB}$ can be expressed as follows:

$$\overrightarrow{PH} \cdot \overrightarrow{AB} = (x - \frac{b\cos\alpha}{c})c^2 + (bc\cos\alpha)y = 0. \tag{2}$$

Similarly, the following equation holds:

$$\overrightarrow{QH} \cdot \overrightarrow{AC} = (bc\cos\alpha)x + (y - \frac{c\cos\alpha}{b})b^2 = 0. \tag{3}$$

Now, using (2) and (3), the following simultaneous equations can be obtained:

$$\begin{aligned} cx - (b\cos\alpha)y &= b\cos\alpha \\ (c\cos\alpha)x + by &= c\cos\alpha. \end{aligned} \tag{4}$$

By solving (4), we can obtain the coefficient *x* as follows:

$$x = \tfrac{b\cos\alpha(b - c\cos\alpha)}{bc\sin^2\alpha}.$$

Using the cosine formula ($b = c\cos\alpha + a\cos\gamma$), *x* is expressed as follows:

$$x = \tfrac{a\cos\alpha\cos\gamma}{c\sin^2\alpha}.$$

In addition, by utilizing the sine formula ($\frac{a}{\sin\alpha} = \frac{c}{\sin\gamma}$), *x* is rewritten as the following equation:

$$x = \tfrac{\cos\alpha\cos\gamma}{\sin\alpha\sin\gamma}.$$

Thus, if we do not consider the case of a right triangle, it is straightforward to rewrite *x* as the following equation:

$$x = \frac{1}{\tan\alpha\tan\gamma} = \frac{\tan\beta}{\tan\alpha\tan\beta\tan\gamma}. \tag{5}$$

Similarly, the coefficient *y* can be represented as follows:

$$y = \frac{\tan\gamma}{\tan\alpha\tan\beta\tan\gamma}. \tag{6}$$

Using the addition theorems of the trigonometric function, we can obtain the following result:

$$\tan\alpha\tan\beta\tan\gamma = \tan\alpha + \tan\beta + \tan\gamma.$$

Thus, (5) and (6) are rewritten as follows:

$$\begin{aligned} x &= \tfrac{\tan\beta}{\tan\alpha+\tan\beta+\tan\gamma} \\ y &= \tfrac{\tan\gamma}{\tan\alpha+\tan\beta+\tan\gamma}. \end{aligned} \tag{7}$$

With respect to a reference point O, (1) can be rewritten as follows:

$$\overrightarrow{AH} = \overrightarrow{OH} - \overrightarrow{OA} = x(\overrightarrow{OB} - \overrightarrow{OA}) + y(\overrightarrow{OC} - \overrightarrow{OA}).$$

Now $\overrightarrow{OH}$ can be represented in the following form:

$$\overrightarrow{OH} = (1 - x - y)\overrightarrow{OA} + x\overrightarrow{OB} + y\overrightarrow{OC}. \tag{8}$$

Substituting (7) into (8), finally, $\overrightarrow{OH}$ is given by

$$\overrightarrow{OH} = \frac{\tan\alpha\overrightarrow{OA} + \tan\beta\overrightarrow{OB} + \tan\gamma\overrightarrow{OC}}{\tan\alpha + \tan\beta + \tan\gamma}. \tag{9}$$

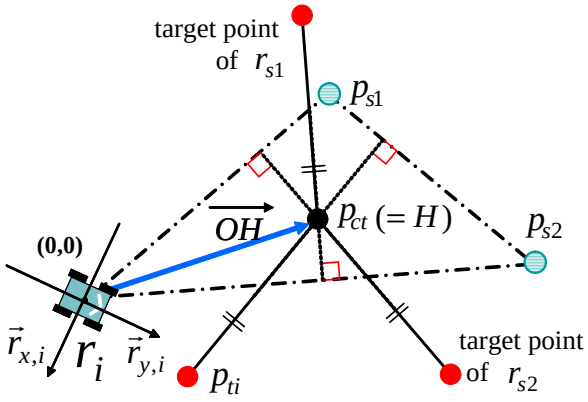


FIGURE 5. Illustrating two important properties: p_{ct} and position vector

Under the adaptive triangular mesh generation algorithm, there are two mathematical properties. First, p_{ti} is determined with p_{ct} in $\mathbb{T}_i(t)$ at t used as H at $t+1$. Moreover, r_i uses p_{ct} at t as a basis to generate $\mathbb{T}_i(t+1)$ with d_r from p_{ct} to p_{ti} at $t+1$ (see Fig. 3-(c)). Secondly, since p_{ct} and H between t and $t+1$ remain unchanged, $\overrightarrow{OH}$ in (9) is the position vector toward p_{ct} with respect to the origin of r_i's local coordinates as illustrated in Fig. 5.

Motion Control. Under the adaptive triangular mesh generation algorithm, three neighboring robots attempt to cooperatively configure themselves into an equilateral triangle, adapting to the density of contamination. As presented in Fig. 6, let's consider the circumscribed an equilateral triangle $\triangle p_{ti}p_{s1}p_{s2}$ configured from three positions occupied by r_i, r_{s1}, and r_{s2} where the center is p_{ct} and the radius is d_a $(= k_a \times \frac{d_u}{\sqrt{3}})$ in length. From the triangular configuration and the two mathematical properties above, we design the motion of r_i by controlling the distance d_i from p_{ct} and the internal angle θ_i between $\overline{p_{ct}p_{ti}}$ and $\overline{p_{ct}p_{s2}}$.

First, d_i in Fig. 6 is controlled by the following equation:

$$\dot{d}_i(t) = -a(d_i(t) - d_a), \tag{10}$$

where a is a positive constant. Indeed, the solution of (10) is obtained as follows:

$$d_i(t) = |d_i(0)|e^{-at} + d_a.$$

The solution converges exponentially to d_a as t approaches infinity. Secondly, θ_i is controlled by the following equation:

$$\dot{\theta}_i(t) = k(\theta_{s1}(t) + \theta_{s2}(t) - 2\theta_i(t)), \tag{11}$$

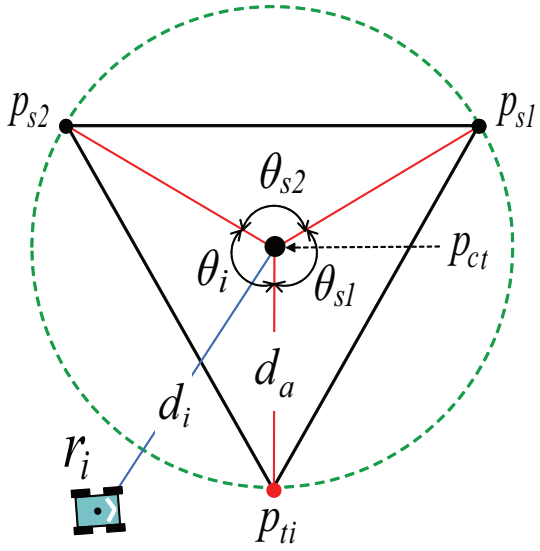


FIGURE 6. Motion control of a robot by the algorithm

where k is a positive number. Using the features of a triangle whose total external angles is 2π, (11) can be rewritten as

$$\dot{\theta}_i(t) = k'(\frac{2}{3}\pi - \theta_i(t)), \tag{12}$$

where k' is $3k$. Similarly, the solution of (12) is obtained as follows:

$$\theta_i(t) = |\theta_i(0)|e^{-k't} + \tfrac{2}{3}\pi.$$

The soultion converges exponentially to $\frac{2}{3}\pi$ as t approaches infinity. Note that (10) and (12) imply that the trajectory of r_i converges to an equilibrium state $[d_a\ \frac{2}{3}\pi]^T$. From Fig. 6, this also implies that the equilibrium for θ_i is defined as $\theta_i = \theta_{s1}$ since $\triangle p_{ti}p_{ct}p_{s1}$ and $\triangle p_{ti}p_{ct}p_{s2}$ are eventually congruent [**16**]. In order to show the convergence into the state $[d_i(t)\ \theta_i(t)]^T$, we will take advantage of stability based on Lyapunov's theory [**31**]. The convergence into the desired stable configuration is one that minimizes the energy level of the scalar function. Consider the following scalar function:

$$f_i(d_i, \theta_i, \theta_{s1}) = \frac{1}{2}(d_i - d_a)^2 + \frac{1}{2}(\theta_{s1} - \theta_i)^2. \tag{13}$$

This scalar function is always positive definite except when $d_i \neq d_a$ and $\theta_i \neq \theta_{s1}$. The derivative of the scalar function is given as follows:

$$\dot{f}_i = -(d_i - d_a)^2 - (\theta_{s1} - \theta_i)^2,$$

which is negative definite. The scalar function is radially unbounded, since it tends to infinity as $||[d_i(t)\ \theta_i(t)]^T|| \to \infty$. Therefore, the equilibrium state is asymptotically stable, implying that r_i reaches a vertex of the stable triangle.

Now we show the convergence of the algorithm for n robots. The n-order scalar function **F** is defined as

$$\mathbf{F} = \sum_{i=1}^{n} f_i(d_i, \theta_i, \theta_{s1}). \tag{14}$$

It is straightforward to verify that **F** is positive definite and $\dot{\mathbf{F}}$ is negative definite. **F** is radially unbounded, since it tends to infinity as t approaches infinity. Consequently, n robots move toward the equilibrium state.

5. Simulation Results and Discussion

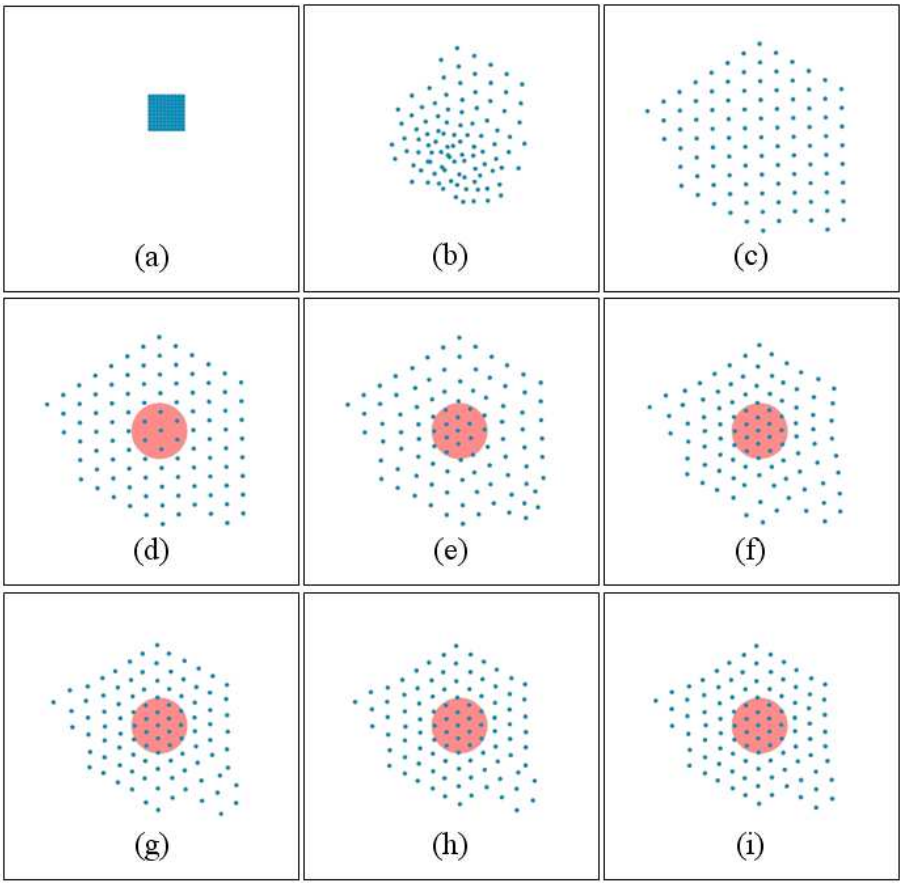
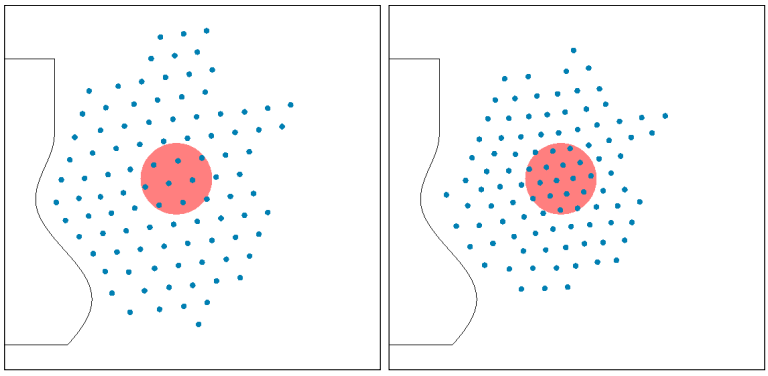


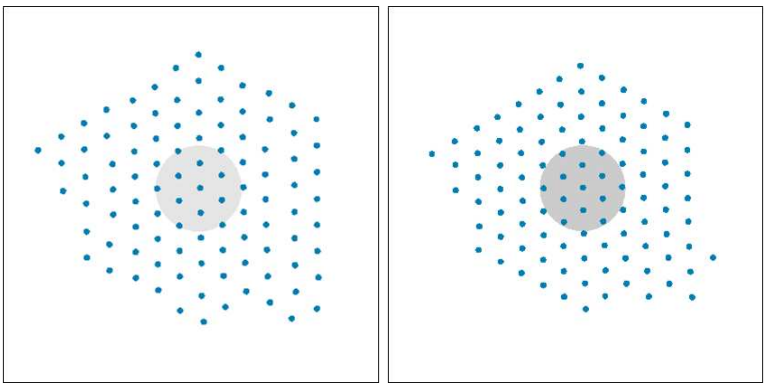
FIGURE 7. Simulation result for the whole deployment of 100 robots ((a)∼(c): uniform configuration [**16**], (d)∼(i): adaptive configuration with increasing center density of 0.7)

In this section, we describe simulations of filling an area of interest with a swarm of robots in order to show the validity of our proposed algorithm. In these simulations, we represent the area of varying contamination density as a colored circle that will be sparsely or densely populated with robots.

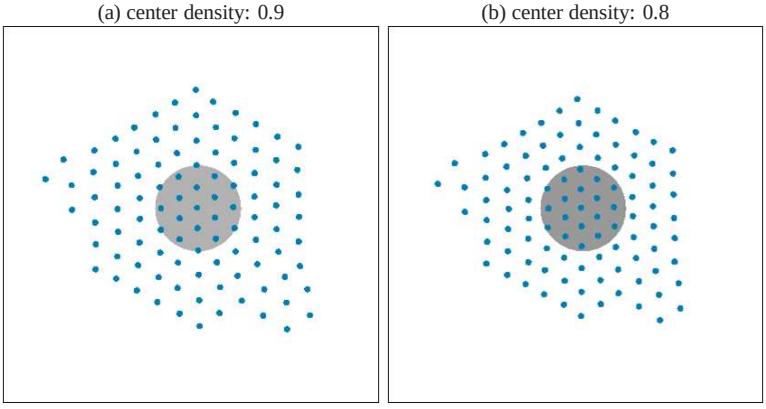


(a) uniform configuration (b) adaptive configuration

FIGURE 8. Adaptive triangular mesh generation with geographic borders



(a) center density: 0.9 (b) center density: 0.8



(c) center density: 0.7 (d) center density: 0.6

FIGURE 9. Simulation results according to varying contamination densities

First, in order to assist in understanding the self-configuration of robot swarms from an initial distribution, Fig. 7 presents snapshots for the simulation result by 100 robots. From the initial distribution in Fig. 7-(a), the robots configured themselves in the 2-dimensional plane (see Fig. 7-(c)). After constructing an equilateral triangle network, we investigated

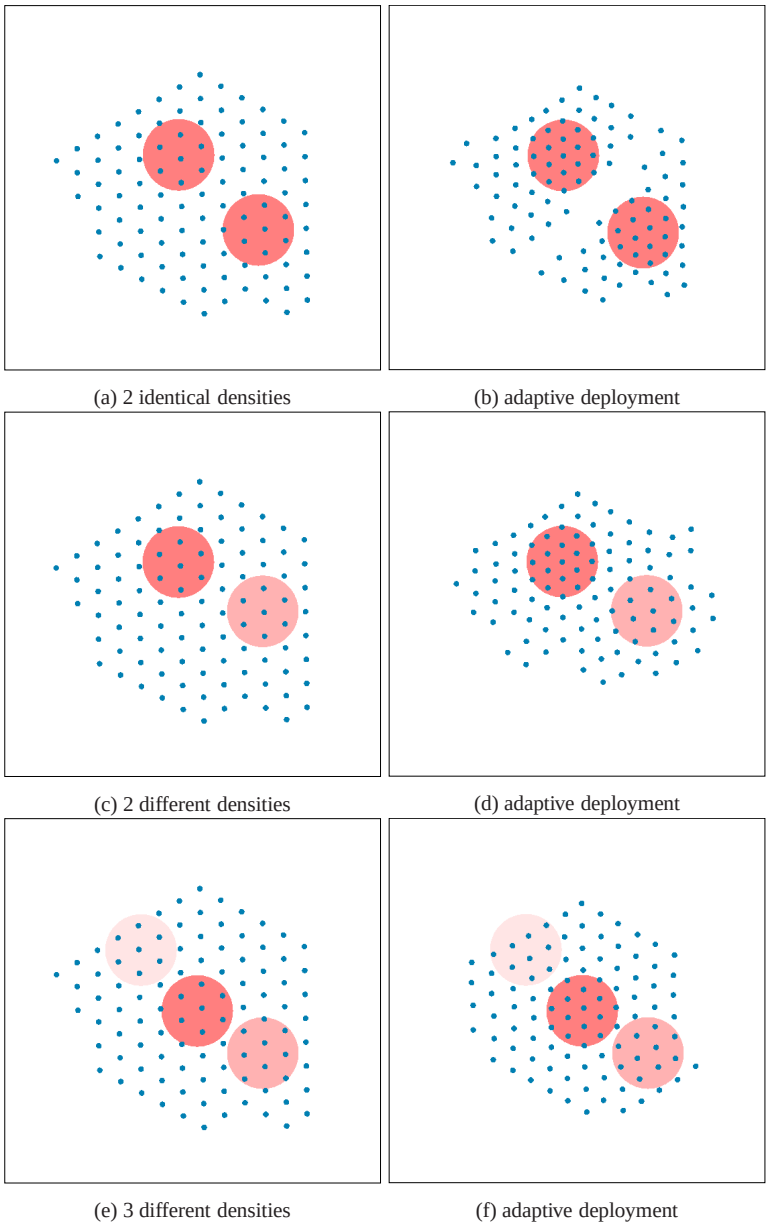
how they adapt their triangular mesh network according to the assigned contamination densities. The snapshots from Fig. 7-(d) to (i) show that robots could generate adaptive triangular meshes based on k_a and d_a. If the series of snapshots are carefully observed, the number of robots within the area of interest, represented by a red circle, increases. In detail, it is initially observed in Fig. 7-(d) that there are 7 robots within the area. By executing the adaptive triangular mesh generation algorithm repeatedly, the number of robots located in that area rises to 17 robots in the final distribution (see Fig. 7-(i)). Compared with Fig. 7-(c), the overall size of the final distribution was reduced. Similarly, Fig. 8 shows the result from another simulation with geographical borders [**17**].

Secondly, Fig. 9 shows the results from simulations with four different degrees of center density. In the figure, the number of robots within the circle increases in proportion to the degree of density. Similarly, the size of the swarm in its final converged shape varies according to the center density. Higher densities forced the robots to decrease the intervals between neighboring robots.

Thirdly, we performed simulations for multiple varying densities in a single swarm. Figs. 10-(a) and (b) show the results for two identical center densities. Although the swarm was spilt into two smaller groups, robots could adaptively configure themselves near the area. Figs. 10-(c) and (d) present the results when the center contamination densities vary: 0.7 on the left hand side and 0.9 on the right hand side. The higher center contamination density area is more densely populated. In Figs. 10-(e) and (f), the simulation was performed for three different center contamination densities in a swarm: 0.9, 0.7, and 0.8 from left to right. Robots could adapt the interval between each other according to the varying contamination densities, which is similar to previous simulations. Thus the overall size of the swarm also varies accordingly.

Three main features highlight our adaptive triangular mesh generation as follows. First, the approach enables the swarm of robots to deploy themselves while adapting to contamination densities. More specially, we proposed motion control to form equilateral triangles with a appropriate intervals depending on the density. Secondly, the triangle lattice is built based on a partially-connected mesh topology since each robot locally interacts with its selected neighbors. Among all the possible types of regular polygons, the triangle element is easy to construct and highly scalable as the number of robots increases. Thirdly, our approach eliminates major assumptions such as robot identifiers, common coordinates, global orientation, specific leaders, and direct communication. Robots calculate their target position without having to remember past actions or states, which makes it easier to cope with transient error.

We believe that our algorithm works well under real world conditions, but several issues remain to be addressed. Our approach relies on the assumption that robots can sense the positions of neighboring robots and contamination densities within *SB*. Practically speaking, it is difficult to precisely measure the positions of other robots using infrared [**9**] [**29**] or sonar sensors [**19**], and to detect the contamination densities. When direct communications are employed, it is possible to exchange information about densities. So, as our future work, (to further facilitate implementation of the proposed method in a real environment,) robot swarms exchanging information are expected to be effectively applied to such deployment. Robots, however, still require *a priori* knowledge, such as individual identifiers or global coordinates. Direct communication may also involve difficulties such as limited bandwidth, range, and interference. These important engineering issues are left for future work.



(a) 2 identical densities (b) adaptive deployment

(c) 2 different densities (d) adaptive deployment

(e) 3 different densities (f) adaptive deployment

FIGURE 10. Simulation results for varying local contamination densities

6. Conclusions

The adaptive triangular mesh generation problem was addressed to disperse a swarm of mobile robots which can adapt to the degree of contamination in a target area. There were several major assumptions underlying our proposed approach to this problem: no robot identifiers, no common coordinates or global orientation, and no direct communication. Robots computed their target positions without requiring memories of past actions or states. Under such conditions, the proposed adaptive mesh generation algorithm enables a

large-scale swarm of robots to configure themselves into triangular patterns, while changing the uniform interval according to the contamination densities that can be detected by sensors. We took advantage of the fact that, among all the possible types of n-polygons, the triangle is highly scalable, and less influenced by the number of neighboring robots. To form the triangular pattern, robots were allowed to interact with only two selected neighbors at each time. By collecting such a local behavior of each robot, a swarm of robots arranged in triangular meshes was self-configured into the area of varying degrees of the local density. The properties of the algorithm were shown mathematically, and also verified through extensive simulations. Finally, we expect that the proposed approach can be used as a simple and effective way to deploy mobile sensor networks for coverage of unknown areas of interest.

Bibliography

[1] Choset, H., "Coverage for robotics-a survey of recent results". Annals of Mathematics and Artificial Intelligence **31**(1-4):113-126 (2001).

[2] Cortes, J., Martinez, S., Karatas, T., and Bullo, F., "Coverage control for mobile sensing networks". IEEE Transactions on Robotics and Automation **20**(2):243-255 (2004).

[3] Suzuki, I. and Yamashita, M., "Distributed anonymous mobile robots: formation of geometric patterns". SIAM Journal of Computing, **28**(4):1347-1363 (1999).

[4] Cao, Z., Tan, M., Wang, S., Fan, Y., and Zhang, B., "The optimization research of formation control for multiple mobile robots". Proc. 4th World Cong. Intelligent Control and Automation, pp.1270-1274 (2002).

[5] Ikemoto, Y., Hasegawa, Y., Fukuda, T., and Matsuda, K., "Graduated spatial pattern formation of robot group". Information Sciences **171**(4):431-445 (2005).

[6] Shimizu, M., Mori, T., and Ishiguro, A., "A development of a modular robot that enables adaptive reconfiguration". Proc. IEEE/RSJ Int. Conf. Intelligent Robots and Systems, pp.174-179 (2006).

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

[8] Howard, A., Mataric, M.J., and Sukhatme, G.S., "Mobile sensor network deployment using potential fields: a distributed, scalable solution to the area coverage problem". Proc. 6th Int. Symp. Distributed Autonomous Robotic Systems, pp.299-308 (2002).

[9] Spears, W., Spears, D., Hamann, J., and Heil, R., "Distributed, physics-based control of swarms of vehicles". Autonomous Robots **17**(2-3):137-162 (2004).

[10] Fujibayashi, K., Murata, S., Sugawara, K., and Yamamura, M., "Self-organizing formation algorithm for active elements". Proc. 21st IEEE Symp. Reliable Distributed Systems, pp.416-421 (2002).

[11] Reif, J. and Wang, H., "Social potential fields: a distributed behavioral control for autonomous robots". Robotics and Autonomous Systems **27**(3):171-194 (1999).

[12] Zheng, Y.F. and Chen, W., "Mobile robot team forming for crystallization of protein". Autononomous Robots **23**(1):69-78 (2007).

[13] Martison, E. and Payton, D. "Lattice formation in mobile autonomous sensor arrays" in *Swarm Robotics* (*LNCS*). Sahin, E. and Spears, W.M. (eds.), **3342**:98-111, Springer, (2005).

[14] McLurkin, J. and Smith, J., "Distributed algorithms for dispersion in indoor environments using a swarm of autonomous mobile robots". Proc. 7th Int. Symp. Distributed Autonomous Robotic Systems, pp.831-840 (2004).

[15] Shucker, B., Murphey, T.D., and Bennett, J.K., "Convergence-preserving switching for topology-dependent decentralized systems". IEEE Transactions on Robotics **24**(6):1405-1415 (2008).

[16] Lee, G. and Chong, N.Y., "A geometric approach to deploying robot swarms". Annals of Mathematics and Artificial Intelligence **52**(2-4):257-280 (2008).

[17] Lee, G. and Chong, N.Y., "Self-configurable mobile robot swarms with hole repair capability". Proc. IEEE/RSJ Int. Conf. Intelligent Robots and Systems, pp.1403-1408 (2008).

[18] Fredslund, J. and Mataric, M.J., "A general algorithm for robot formations using local sensing and minimal communication". IEEE Transactions on Robotics and Automation **18**(5):837-846 (2002).

[19] Lee, G. and Chong, N.Y., "Decentralized formation control for small-scale robot teams with anonymity". Mechatronics **19**(1):85-105 (2009).

[20] Folino, G. and Spezzano, G., "An adaptive flocking algorithm for spatial clustering" in *Parallel Problem Solving from Nature - PPSN VII* (*LNCS*). Goos, G., Hartmanis, J., and Leeuwen, J.V. (eds.), **2439**:924-933, Springer,(2002).

[21] Lee, G. and Chong, N.Y., "Adaptive flocking of robot swarms: algorithms and properties". IEICE Transactions on Communications **E91-B**(9):2848-2855 (2008).

[22] Fax, J.A. and Murray, R.M., "Information flow and cooperative control of vehicle formations". IEEE Transactions on Automatic Control **49**(9):1465-1476 (2004).

[23] Olfati-Saber, R. and Murray, R.M., “Consensus problems in networks of agents with switching topology and time-delays”. IEEE Transactions on Automatic Control **49**(9):1520-1533 (2004).

[24] Burgard, W., Moors, M., Stachniss C., and Schneider, F., “Coordinated multi-robot exploration”. IEEE Transactions on Robotics and Automation **20**(3):120-145 (2005).

[25] Mondada, F., Pettinaro, G.C., Guignard, A., Kwee, I., Floreano, D., Deneubourg, J.-L., Nolfi, S., Gambardella, L.M., and Dorigo, M., “Swarm-bot: a new distributed robotic concept”. Autonomous Robots **17**(2-3):193-221 (2004).

[26] Jatmiko, W., Sekiyama, K., and Fukuda, T., “A particle swarm-based mobile sensor network for odor source localization in a dynamic environment” in *Distributed Autonomous Robotic Systems 7*. Gini, M. and Voyles, R. (eds.), Springer Japan, pp.71-80 (2007).

[27] Jung, B. and Sukhatme, G.S., “Tracking targets using multiple robots: the effect of environment occlusion”. Autonomous Robots **13**(3):191-205 (2002).

[28] Krishnanand, K.N., Amruth, P., and Guruprasad, M.H., “Glowworm-inspired robot swarm for simultaneous taxis towards multiple radiation sources”. Proc. IEEE Int. Conf. Robotics and Automation, pp.958- 963 (2006).

[29] Lee, G., Yoon, S., Chong, N.Y., and Christensen, H., “Self-configuring robot swarms with dual rotating infrared sensors”. Proc. IEEE/RSJ Int. Conf. Intelligent Robots and Systems, pp.4357-4362 (2009).

[30] Ghosh, S., Basu, K., and Das, S.K., “An architecture for next-generation radio access networks”. IEEE Network **19**(5):35-42 (2005).

[31] Slotine, J.E. and Li, W., *Applied nonlinear control*. Prentice-Hall (1991).