Vol.2015-AL-155 No.17
2015/11/21

IPSJ SIG Technical Report

Formation problems for synchronous mobile robots in the three dimensional Euclidean space

Yukiko Yamauchi[1,a)] Taichi Uehara[1,b)] Masafumi Yamashita[1,c)]

Abstract: We consider a swarm of autonomous mobile robots each of which is anonymous and oblivious (memoryless), and synchronously executes the same algorithm. The *plane formation problem* requires the robots to land on a common plane. The *pattern formation problem* requires the robots to form a given target pattern. We investigate these two formation problems for oblivious fully-synchronous (FSYNC) robots moving in the three dimensional Euclidean space (3D-space), and characterize these problems by showing necessary and sufficient conditions for the robots to achieve these formation tasks by using the notion of *symmetricity* of the positions of robots in 3D-space. For solvable instances of the pattern formation problem, we give a distributed algorithm for oblivious FSYNC robots.

1. Introduction

Distributed control of a system consisting of autonomous mobile computing entities in the three dimensional Euclidean space (3D-space) is one of the most challenging problems in distributed computing theory and robotics. One of the most important properties that is expected for such systems is *self-organization ability* that enables the system to obtain the coordination by itself. For example, drones are becoming widely available and their applications in sensing, monitoring, and rescues in harsh environment such as disaster area and active volcanoes, where they are required to coordinate themselves without human intervention, are attracting much attention.

As one of the fundamental tasks for robots in 3D-space, we consider two formation problems. The *plane formation problem* requires that a set of robots land on a common plane, that is not predefined, without making any multiplicity. The *pattern formation problem* requires that a set of robots form a given 3D target pattern, that is given as the set of coordinates of points. A robot is a point in 3D-space that autonomously moves according to a given rule. Specifically, each robot repeats a Look-Compute-Move cycle, where it observes the positions of other robots (Look phase), computes its next position with a given algorithm (Compute phase), and moves to the next position (Move phase). Each robot is *anonymous* in the sense that they have no identifier and the robots are *uniform* in the sense that all robots execute the same algorithm. Each robot has no access to the global x-y-z coordinate system and in a Look phase it observes the positions of other robots in its *local x-y-z coordinate system*. The origin of the local coordinate system of a robot is its current position and the local coordinate system has arbitrary directions and unit distance. However we assume that all local coordinate systems are right-handed. In other words, the robots have *chirality*. Each robot is *oblivious* in the sense that in each Compute phase, the input to the common algorithm is the observation of the current phase and the robot does not remember the past. In a Move phase, each robot reaches the computed destination, i.e., its movement is *rigid*. We consider the *fully-synchronous (FSYNC)* model where the robots execute the t-th Look-Compute-Move cycle at the same time with each of the Look, Compute, and Move phases completely synchronized. Here the *configuration* of robots is the positions of robots observed in the global coordinate system, i.e., a multiset of points. These assumptions mean that the robots do not have explicit communication medium and they have to tolerate inconsistency among local coordinate systems. The robots coordinate themselves by just observing the positions of other robots and building some agreement on some reference points or some common coordinate system.

The pattern formation problem was first introduced by Suzuki and Yamashita for the robots moving on the two-dimensional Euclidean space (2D-space) [8]. They characterized the class of formable patterns by using the notion of *symmetricity* of an initial configuration. The symmetricity of a configuration is essentially its rotational symmetry. Let P be a configuration of robots in 2D-space, i.e., a set of points. We consider the decomposition of P into regular m-gons centered at the center of the smallest enclosing circle of P. The symmetricity $\rho(P)$ of P is the maximum value of such m. Here we consider a point as a regular 1-gon with an arbitrary center and a set of two points as a regular 2-gon with the center at the midpoint of the two points. When there is a robot on the center of the smallest enclosing circle of P, $\rho(P) = 1$. Then they showed that oblivious FSYNC robots can form a target pattern F from a given initial configuration P if and only if $\rho(P)$ divides $\rho(F)$. This notion of symmetricity is based on the fact that there exists an arrangement of local coordinate systems

[1] Kyushu University, 744 Motooka, Nishi-ku, Fukuoka 819-0395, Japan
[a)] yamauchi@inf.kyushu-u.ac.jp
[b)] uehara@tcslab.csce.kyushu-u.ac.jp
[c)] mak@inf.kyushu-u.ac.jp

of robots that keeps the robots forming regular $\rho(P)$-gons forever. Let P' be one of the $\rho(P)$-gons of P. Then, we pick up one robot (and its local coordinate system) from P' and apply the rotations by $2k\pi/\rho(P)$ ($k = 1, 2, \cdots, \rho(P)$) around the center of the smallest enclosing circle of robots. This defines cyclic local coordinate systems of robots forming P' and because they all have the same local observation and execute the same algorithm, their next locations form a regular $\rho(P)$-gon with the same center. The robots are FSYNC and they all move to these symmetric next locations. In this way, the robots never break regular polygons.

Yamauchi et al. focus on the fact that the above worst case in 2D-space is caused by the rotations that form a *cyclic group* and extended this impossibility result in 2D-space to 3D-space by using the *rotation groups* when they consider the plane formation problem [10]. The rotation groups in 3D-space are defined by a set of rotation axes and their arrangement. There are five kinds of finite-order rotation groups; *the cyclic groups*, *the dihedral groups*, *the tetrahedral group*, *the octahedral group*, and *the icosahedral group* [1], [2]. Given a configuration P of robots in 3D-space, its rotation group $\gamma(P)$ is the rotation group applicable to P and none of its supergroup in these five kinds of rotation groups is applicable to P, which of course is uniquely determined. The rotation group $\gamma(P)$ decomposes P into *vertex-transitive* set of points by its group action on P. A set of points Q is vertex-transitive regarding a rotation group G if (i) for any $q, q' \in Q$, there exists an element $g \in G$ such that $q = g * q'$ and (ii) for each $g \in G$ and for each $q \in Q$, $g * q \in Q$. When a set of positions of robots is vertex-transitive, the robots may have the same local observation. Yamauchi et al. called the cyclic groups and the dihedral groups *two-dimensional (2D) groups* because they have the single (principal) rotation axis that is easily recognized by the robots. Then the robots can easily land on the plane that is perpendicular to the recognized rotation axis and contains the center of the smallest enclosing ball of robots. On the other hand, they called remaining three rotation groups *three-dimensional (3D) groups*, because these three rotation groups do not have such distinguishable single rotation axis and when the rotation group of a configuration is 3D, the robots are not on one plane. Then they showed that the robots cannot form a plane from a given configuration P if and only if $\gamma(P)$ is a 3D group and all robots are not on the rotation axis of $\gamma(P)$. The intuition behind this impossibility is completely same as that in 2D-space, i.e., we can generate symmetric local coordinate systems regarding $\gamma(P)$ and the robots never break the vertex-transitive polyhedra regarding $\gamma(P)$ if none of the robots are on the rotation axis.

In this paper, we define the *symmetricity* of a configuration of robots in 3D-space as a set of rotation groups that acts on the positions of robots and consists of rotation axes containing no robot. For a set of points P, we denote its symmetricity by $\varrho(P)$. Then we will show the following theorem.

Theorem 1 Oblivious FSYNC robots can form a target pattern F from an initial configuration P if and only if $\varrho(P) \subseteq \varrho(F)$.

The impossibility is caused by symmetric arrangement of local coordinate systems. For the solvable cases, we present a pattern formation algorithm for oblivious FSYNC robots.

By using the notion of symmetricity, we can rephrase the necessary and sufficient condition of the plane formation problem [10] as follows:

Theorem 2 Oblivious FSYNC robots can form a plane from an initial configuration P if and only if $\varrho(P)$ consists of 2D-groups.

Related work. The pattern formation problem has been investigated for the robots in 2D-space and little is known for the robots in 3D-space. In the following, we briefly survey existing results of the pattern formation problem in 2D-space.

The two classical synchrony models are the *fully-synchronous (FSYNC) model* and the *semi-synchronous (SSYNC) model* [8]. In the FSYNC model, the robots execute the t-th Look-Compute-Move cycle with each of the Look, Compute, and Move phases completely synchronized for any t. In the SSYNC model, the executions of Look-Compute-Move cycles are synchronized, but some robots may not execute some cycles. The *asynchronous (ASYNC)* model, which was first introduced by Flocchini et. al. [5], puts no assumption on the execution of cycles.

Dieudonné et al. showed that for more than three oblivious ASYNC robots, the leader election problem is solvable from P, if and only if every pattern F is formable from P, provided that F does not contain multiplicities [4]. Suzuki and Yamashita first investigated the impact of availability of memory at robots and synchrony among robots on the set of formable patterns for the SSYNC and the FSYNC models [8], [9]. Fujinaga et al. extended the result to the ASYNC model [7]. It has been shown that (i) non-oblivious FSYNC robots can form a target pattern F from an initial configuration P if and only if $\rho(P)$ divides $\rho(F)$, and (ii) oblivious ASYNC (SSYNC) robots can form a target pattern F from an initial configuration P if and only if $\rho(P)$ divides $\rho(F)$ except the case where F is a point of multiplicity 2. The exceptional case is called the *rendezvous problem* or the *gathering problem* for two robots. However, the gathering problem for more than two robots is solvable by oblivious SSYNC robots [8] and ASYNC robots [3]. These results assume that the robots have the *weak multiplicity detection ability*, i.e., they can check whether a point is occupied by one robot or more than one robot. Without weak multiplicity detection ability, the gathering problem becomes unsolvable.

For most of the results known for mobile robots up to year 2012, see the book by Flocchini et al. [6].

Organization. In Section 2, we define the mobile robot model and we introduce the rotation groups and symmetricity of a configuration in Section 3. We prove the necessity of Theorem 1 in Section 4 and sufficiency of Theorem 1 in Section 5 by showing a pattern formation algorithm for oblivious FSYNC robots. We then discuss the results of [10] and show Theorem 2 in Section 6. We conclude this paper with Section 7.

2. Preliminary

Let $R = \{r_1, r_2, \ldots, r_n\}$ be a set of $n \geq 3$ robots each of which is represented by a point in 3D-space. Each robot is anonymous and there is no way to distinguish them. We use the indexes just for description.

By Z_0 we denote the global x-y-z coordinate system. Let $p_i(t) \in \mathbb{R}^3$ be the position of r_i at time t in Z_0, where $\mathbb{R}$ is the set of real numbers. A *configuration* of R at time t is denoted by a multiset $P(t) = \{p_1(t), p_2(t), \ldots, p_n(t)\}$. We assume that the robots initially occupy distinct positions, i.e., $p_i(0) \neq p_j(0)$ for all $1 \leq i < j \leq n$. [*1] The robots have no access to Z_0. Instead, each robot r_i observes the positions of other robots in its local x-y-z coordinate system Z_i, where the origin is always its current location, while the direction of each positive axis and the magnitude of the unit distance are arbitrary but never change.[*2] We assume that Z_0 and all Z_i are right-handed. In other words, they have *chirality*. By $Z_i(p)$ we denote the coordinates of a point p in Z_i.

We consider discrete time $0, 1, 2, \cdots$ and at each time step the robots executes a Look-Compute-Move cycle with each of Look, Compute, and Move phases completely synchronized, i.e., we investigate the *fully-synchronous (FSYNC)* robots in this paper. We specifically assume without loss of generality that the $(t+1)$-th Look-Compute-Move cycle starts at time t and finishes before time $t+1$. At time t, each $r_i \in R$ obtains a multiset $Z_i(P(t)) = \{Z_i(p_1(t)), Z_i(p_2(t)), \ldots, Z_i(p_n(t))\}$. We call $Z_i(P(t))$ the *local observation* of r_i at t. Then r_i computes its next position using an algorithm ψ, which is common to all robots. If ψ uses only $Z_i(P(t))$, we say that r_i is *oblivious*. Otherwise, we say r_i is *non-oblivious*, i.e., r_i can use past local observations and past outputs of ψ. Formally, ψ is a total function from $\mathcal{P}_n^3$ to $\mathbb{R}^3$, where $\mathcal{P}_n^3 = (\mathbb{R}^3)^n$ is the set of all configurations (which may contain multiplicities). Finally, r_i moves to $\psi(Z_i(P(t)))$ in Z_i before time $t+1$. Thus the movement of robots is *rigid*. In this paper, we do not care for the track of the movement of robots, rather each robot jumps to its next position. An infinite sequence of configurations $\mathcal{E} : P(0), P(1), \ldots$ is called an *execution* from an *initial configuration* $P(0)$. Observe that the execution $\mathcal{E}$ is uniquely determined, once initial configuration $P(0)$, local coordinate systems of robots at time 0, and algorithm ψ are fixed.

The *plane formation problem* is to make the robots land on a common plane without making any multiplicity from a given initial configuration.

The *pattern formation problem* is to make the robots form a given target pattern F from a given initial configuration. The target pattern F is given to each robot as a multiset of coordinates of n points in Z_0. We assume that F does not contain any multiplicity. Because robots do not have access to the global coordinate system, it is impossible to form F itself. Let $\mathcal{T}$ be the set of all rotations, translations, and uniform scalings. We say F' is *similar* to F if there exists $Z \in \mathcal{T}$ such that $F' = Z(F)$, which we denote by $F' \simeq F$. We say that the robots form a target pattern F from an initial configuration P, if, regardless of the choice of initial local coordinate systems of robots in the initial configuration, the execution $P(0)(= P), P(1), \ldots$ reaches a configuration $P(t)$ that is similar to F in finite time.

*1 This assumption is necessary because it is impossible to break up multiple oblivious FSYNC robots (with the same local coordinate system) on a single position as long as they execute the same algorithm. The proposed pattern formation algorithm does not make any multiplicity during the formation. However, we have to consider configurations with multiplicity when we prove impossibility by checking executions of any arbitrary algorithm.

*2 Since Z_i changes whenever r_i moves, notation $Z_i(t)$ is more rigid, but we omit parameter t to simplify its notation.

Table 1 Three polyhedral groups. The number of elements around k-fold axes except the identity element is shown. The number in the bracket is the number of rotation axes with that fold.

Polyhedral group	Fold of rotation axes				Order
	2	3	4	5	
T	3(3)	8(4)	-	-	12
O	6(6)	8(4)	9(3)	-	24
I	15(15)	20(10)	-	24(6)	60

For any (multi)set of points P, by $B(P)$ and $b(P)$, we denote the *smallest enclosing ball* of P and its center, respectively. A point on the sphere of a ball is said to be *on* the ball, and we assume that the *interior* or the *exterior* of a ball does not include its sphere. The *largest empty ball* $L(P)$ of P is the ball centered at $b(P)$ and contains no point of P in its interior, but contains at least one point on it.

3. Symmetricity in 3D-Space

In this section, we define the *rotation group* and the *symmetricity* of a set of points, and related notions.

We formally define the five kinds of rotation groups. The rotation group $SO(3)$ has five subgroups of finite order [1], [2]; the cyclic group C_k ($k = 1, 2, \cdots$), the dihedral group D_ℓ ($\ell = 2, 3, \cdots$), the tetrahedral group T, the octahedral group O, and the icosahedral group I. Each of these groups is identified by a group formed by the rotations of a regular pyramid, a regular prism, a regular tetrahedron, a regular octahedron, and a regular icosahedron, respectively. For example, consider a regular pyramid that has a regular k-gon as its base. The rotation operations for this regular pyramid are rotations by $2\pi i/k$ for $i = 1, 2, \cdots, k$ around an axis containing the apex and the center of the base. We call such an axis *k-fold axis*. Let a^i be the rotation by $2\pi i/k$ around this k-fold axis with $a^k = e$ where e is the identity element. Then, $a^1, a^2, \ldots, a^k$ form the *cyclic group* C_k.

A regular prism (except a cube) has two parallel regular k-gons as its top and bottom bases and has two types of rotation axes, one is the k-fold axis containing the centers of its top and bottom bases, and the others are k 2-fold axes that exchange the top and the bottom. We call this k-fold axis *principal axis*, and the remaining k 2-fold axes *secondary axes*. These rotation operations on a regular prism form the *dihedral group* D_k. When $k = 2$, we can define D_2 in the same way. Though in the group theory we do not distinguish the principal axis, when we consider the rotation group of a set of points, we can show that we can distinguish the principal axis from the three 2-fold rotation axes of D_2.

The remaining three rotation groups T, O, and I are called the polyhedral groups. Each of the three rotation groups consists of rotation axes containing the midpoints of two opposite edges, those containing the centers of opposite faces, and those containing opposite vertices. Table 1 shows the number of rotation axes and the number of elements around each type of rotation axes for each of the polyhedral groups.

Let $\mathbb{S} = \{C_k, D_\ell, T, O, I \mid k = 1, 2, \ldots, and\ \ell = 2, 3, \ldots\}$ be the set of rotation groups, where C_1 is the rotation group with order 1; its unique element is the identity element (i.e., 1-fold rotation). We denote the order of $G \in \mathbb{S}$ by $|G|$.

When G' is a subgroup of G ($G, G' \in \mathbb{S}$), we denote it by $G' \preceq G$. If G' is a proper subgroup of G (i.e., $G' \neq G$), we denote it by $G' \prec G$. For example, we have $T \prec O$, $T \prec I$, but $O \not\prec I$. If $G \in \mathbb{S}$ has a k-fold axis, then $C_k \preceq G$. Clearly, $C_{k'} \preceq C_k$ if k' is a divisor of k, which also holds for dihedral groups.

Definition 3 Let P be a set of n points. The rotation group $\gamma(P)$ of a set of points P is the rotation group in $\mathbb{S}$ applicable to P and none of its proper supergroup in $\mathbb{S}$ is applicable to P.

From the definition, $\gamma(P)$ is always uniquely determined irrespective of the local coordinate systems through which a robot observes P. We say a rotation axis of $\gamma(P)$ is *occupied* when it contains some points of P and *unoccupied* otherwise. For example, when P forms a cube, $\gamma(P) = O$ and all 3-fold rotation axes of O are occupied while all 2-fold rotation axes and all 4-fold rotation axes are unoccupied.

Given a set of points P, $\gamma(P)$ determines the arrangement of its rotation axes. We thus use $\gamma(P)$ and its arrangement in P interchangeably. A set of points P can be decomposed into disjoint subsets $\{P_1, P_2, \ldots, P_m\}$ where each element in the family is an orbit of some element of P: For a point $p \in P$, let $Orb(p) = \{g * p \mid g \in \gamma(P)\}$ be the orbit of the group action of $\gamma(P)$ through p. Then we let $\{P_1, P_2, \ldots, P_m\} = \{Orb(p) \mid p \in P\}$ be its orbit space. Such partition is unique and we call it the $\gamma(P)$-*decomposition* of P [10]. Note that the sizes of the elements of the $\gamma(P)$-decomposition of P may be different. Such decomposition does no depend on the local coordinate systems and each robot can recognize it.

For two groups $G, H \in \mathbb{S}$, an *embedding* of G to H is an embedding of each rotation axis of G to one of the rotation axes of H so that any k-fold axis of G overlaps a k'-fold axis of H satisfying $k|k'$ with keeping the arrangement, where $a|b$ represents that a divides b. For example, we can embed T to O so that each 3-fold axis of T overlaps a 3-fold rotation axis of O, and each 2-fold rotation axis of T to a 4-fold axis of O. Note that there may be many embeddings of G to H. Observe that we can embed G to H if and only if $G \preceq H$.

We say that a set of points P is *vertex-transitive* regarding a rotation group G, if (i) for any two points $p, q \in P$, $g * p = q$ for some $g \in G$, and (ii) $g * p \in P$ for all $g \in G$ and $p \in P$, where $*$ denotes the group action. On the other hand, a vertex-transitive set of points is obtained by a rotation group G and a seed point s, i.e., we apply the rotations of G to s. Clearly, each element of the $\gamma(P)$-decomposition of a set of points P is a vertex-transitive set of points regarding $\gamma(P)$. For a vertex-transitive set of points P and any $p \in P$, we call $\mu(p) = |\{g \in G : g * s = p\}|$ the *folding* of p.[*3] We of course count the identity element of G for $\mu(p)$, and $\mu(p) \geq 1$ holds for all $p \in P$. If $p \in P$ is at $b(P)$, its folding is $|\gamma(P)|$ and if p is on a k-fold rotation axis of $\gamma(P)$, its folding is k. Hence $\mu(p)$ for $p \in P$ is identical for a set of points P generated by a rotation group G and a seed point s. We abuse μ to a vertex-transitive set of points P, and $\mu(P)$ represents $\mu(p)$ for $p \in P$. When $\mu(P) > 1$, the positions of points of P is uniquely determined in the arrangement of G if we ignore uniform scalings that keep the center of G. Additionally, we have $|P| = |G|/\mu$. Table 2 shows the set of points generated by the five kinds of rotation groups.

Table 2 The folding of seed points and vertex-transitive sets of points

Group	Order	Folding	Cardinality	Polyhedron
Any G	$\lvert G\rvert$	$\lvert G\rvert$	1	Point
C_k	k	k	1	Point
		1	k	Regular k-gons
D_2	4	2	2	Line
		1	4	Regular tetrahedron, Infinitely many sphenoids, Infinitely many rectangle
D_k	$2k$	k	2	Line
		2	k	Regular k-gon
		1	$2k$	Infinitely many polyhedra
T	12	3	4	Regular tetrahedron
		2	6	Regular octahedron
		1	12	Infinitely many polyhedra
O	24	4	6	Regular octahedron
		3	8	Cube
		2	12	Cuboctahedron
		1	24	Infinitely many polyhedra
I	60	5	12	Regular icosahedron
		3	20	Regular dodecahedron
		2	30	Icosidodecahedron
		1	60	Infinitely many polyhedra

For a set of points P and $G \preceq \gamma(P)$ ($G \in \mathbb{S}$), an arrangement of G (i.e., an embedding of G to $\gamma(P)$) defines a decomposition of P into disjoint subsets by its group action. We call such a decomposition *a G-decomposition* of P.

Definition 4 Let P be a set of n points. The symmetricity $\varrho(P)$ of a set of points P is the set of rotation groups $G \in \mathbb{S}$ that acts on P and there exists an embedding of G to $\gamma(P)$ such that each element of G-decomposition of P is a $|G|$-set.

From the definition, $\varrho(P)$ always contains C_1 and if $G \in \varrho(P)$, $\varrho(P)$ contains every element of $\mathbb{S}$ that is a subgroup of G. When it is clear from the context, we denote the elements of $\varrho(P)$ by the set of rotation groups such that none of its proper supergroup in $\mathbb{S}$ is in $\varrho(P)$. For example, if P forms a cube, $\varrho(P) = \{C_1, C_2, C_4, D_2, D_4\}$, and we denote it by $\varrho(P) = \{D_4\}$.

Because any $G \in \varrho(P)$ acts on P, G is a subgroup of $\gamma(P)$ and any initial configuration P is a set of n points, we can rephrase the above definition as follows: For an initial configuration P, $\varrho(P)$ is the set of rotation groups $G \in \mathbb{S}$ that has an embedding to unoccupied rotation axes of $\gamma(P)$ and if all rotation axes of $\gamma(P)$ is occupied, $\varrho(P) = \{C_1\}$. Because an initial configuration P does not contain any multiplicity, if a point of P is on a rotation axis of $\gamma(P)$, none of $G \in \varrho(P)$ has that axis because such point produces a vertex-transitive set of points with size smaller that $|G|$.

We further introduce the rotation group of local coordinate systems of robots. Of course robots may recognize neither the local coordinate systems of other robots nor the rotation group of local coordinate systems at a glance. This notion is useful when we prove impossibility results.

We denote an arrangement of local coordinate systems by a set of four-tuples $Q = \{(p_i, x_i, y_i, z_i) \mid r_i \in R\}$ where p_i represents the position of $r_i \in R$ in Z_0 and x_i, y_i, z_i are the positions $(1, 0, 0)$, $(0, 1, 0)$, and $(0, 0, 1)$ of Z_i observed in Z_0. An arrangement of local coordinate systems encodes the positions of the robots since the current position of the robot is the origin of its local coordinate system. We also use the set of points $P = \{p_1, p_2, \ldots, p_n\}$ to

[*3] In group theory, the folding of a point P is the size of the stabilizer of p defined by $G(p) = \{g \in G : g * p = p\}$.

describe the positions of robots of Q.[*4]

We consider rotations on Q that produces the same arrangement of local coordinate systems.

Definition 5 The rotation group $\sigma(Q)$ of a set of local coordinate systems Q is the rotation group in $\mathbb{S}$ applicable to Q and none of its proper supergroup in $\mathbb{S}$ is applicable to Q.

Clearly, $\sigma(Q)$ is uniquely determined and it also determines the arrangement of rotation axes of $\sigma(Q)$ in Q that decomposes Q into disjoint subsets by the group action of $\sigma(Q)$. We call this partition $\sigma(Q)$-*decomposition* of Q. We focus on the decomposition of P by $\sigma(Q)$ rather than the decomposition of Q by $\sigma(Q)$. If it is clear from the context, we denote $\sigma(Q)$ by $\sigma(P)$ and the $\sigma(Q)$-decomposition of Q by the set of origins (i.e., positions of robots) of Q.

We finally note that the robots can agree on $\gamma(P)$ irrespective of local coordinate systems, while they cannot agree on $\sigma(P)$ by just observing P in their local coordinate systems.

4. Necessity of Theorem 1

In this section, we prove the necessity of Theorem 1.

Theorem 6 Oblivious FSYNC robots can form a target pattern F from an initial configuration P only if $\varrho(P) \subseteq \varrho(F)$.

We first show the following relations between the rotation group of local coordinate systems and the rotation group of positions of robots.

Property 7 Let P and $\{P_1, P_2, \dots, P_\ell\}$ be a set of n points and its $\sigma(P)$-decomposition. Then we have the following three properties:
(i) $\sigma(P) \preceq \gamma(P)$, thus there is an embedding of $\sigma(P)$ to $\gamma(P)$.
(ii) For each P_i $(1 \leq i \leq \ell)$, the robot forming P_i have the same observation.
(iii) $|P_i| = |\sigma(P)|$ for each $1 \leq i \leq \ell$.

Lemma 8 Let P be an arbitrary initial configuration. For an arbitrary deterministic algorithm ψ and its execution $P(0)(= P), P(1), P(2), \dots$, we have $\sigma(P(t)) \succeq \sigma(P)$ for any $t \geq 0$, thus $\gamma(P(t)) \succeq \sigma(P)$.

Proof. Let P and $\{P_1, P_2, \dots, P_m\}$ be an initial configuration and its $\sigma(P)$-decomposition of P. Because P is a set of n points, from Property 7, $|P_i| = |\sigma(P)|$ for $1 \leq i \leq m$. Let $P(0)(= P), P(1), P(2), \dots$ be the execution of an arbitrary algorithm ψ. We focus on an arbitrary element P_i. Let $p_j \in P_i$. For any $p_k \in P_i$, there exists an element $g_k \in \sigma(P)$ that satisfies $g_k * p_j = p_k$ and $g_k \neq g_{k'}$ if $k \neq k'$. We will show that the movement of each $p_k \in P_i$ is symmetric regarding g_k, hence the robots of P_i keeps the rotation axes of $\sigma(P)$.

Consider the Compute phase at time 0, and let $\psi(Z_j(P(0))) = d_j$ (in Z_j). From Property 7, each robot $p_k \in P_i$ have the same local observation and $\psi(Z_k(P(0))) = \psi(Z_j(P(0))) = d_j$ (in Z_j). Because $p_j, p_k \in P_i$, $Z_k(P(0)) = g_k * Z_j(P(0)) = Z_j(P(0))$. Hence, $\psi(Z_k(P(0))) = d_k = d_j$ (in Z_k). Clearly, we have $Z_k^{-1}(d_k) = g_k * Z_j^{-1}(d_j)$, and after the movement the positions of robots that formed P_i are symmetric regarding the same arrangement of $\sigma(P)$. Additionally, the local coordinate system of these robots are symmetric regarding the same arrangement of $\sigma(P)$.

[*4] Here we assume that the robots occupy distinct positions.

Let $P_i(1) \subseteq P(1)$ be the positions of robots that formed P_i in $P(0)$. Hence, we have $\sigma(P_i(1)) = \sigma(P(0))$. Because this property holds for all P_i $(1 \leq i \leq m)$, we have $\sigma(P(1)) \succeq \sigma(P(0))$. Note that $P(1)$ can be a multiset of points.

We can generate an infinite execution $P(0)(= P), P(1), \cdots$ such that $\sigma(P(t)) \succeq \sigma(P(0))$ by repeating the above procedure to each $P_i(t-1)$ $(i = 1, 2, \cdots, m$ and $t \geq 1)$. Thus $\gamma(P(t)) \succeq \sigma(P(0))$ for all $t \geq 0$. □

We further have the following property.

Lemma 9 For an arbitrary configuration P without multiplicity, $\sigma(P) \in \varrho(P)$.

We now prove Theorem 6.

Proof. (Proof of Theorem 6) Let P and F be a given initial configuration and a target pattern without multiplicity that satisfy $\varrho(P) \not\subseteq \varrho(F)$. Hence, there exists $G \in \varrho(P)$ such that $G \notin \varrho(F)$.

Assume that there exists an algorithm ψ that forms F from P, for contradiction. Assume that $\sigma(P) = G$. From the assumption, there exists at least one execution $P(0)(= P), P(1), P(2), \dots, P(t) \simeq F$. From Lemma 8, we have $\sigma(P(t)) \succeq \sigma(P(0)) = G$.

From the definition of the symmetricity, we have the following: if $G' \in \varrho(F)$, any subgroup of G' is in $\varrho(F)$. In other words, if there exists a subgroup of G' that is not in $\varrho(F)$, then $G' \notin \varrho(F)$. Because $\sigma(P(0)) = G$ is not in $\varrho(F)$, its supergroup $\sigma(P(t)) = \sigma(F)$ is not in $\varrho(F)$.

This contradicts Lemma 9 that says for an arbitrary configuration F without multiplicity, $\sigma(F) \in \varrho(F)$. □

5. Sufficiency of Theorem 1

In this section, we show the sufficiency of Theorem 1.

Theorem 10 Oblivious FSYNC robots can form a target pattern F from an initial configuration P if $\varrho(P) \subseteq \varrho(F)$.

We present a pattern formation algorithm ψ_{PF} that makes oblivious FSYNC robots form a target pattern F from a given initial configuration P if P and F satisfies the condition of Theorem 10. The proposed algorithm consists of three phases: The first phase translates P into another configuration P' that satisfies $\gamma(P') \in \varrho(P)$ and no robot is on the rotation axis of $\gamma(P')$ in P'. The second phase makes the robots agree on an embedding of F into P' so that the embedded target pattern $\widetilde{F}$ satisfies $b(\widetilde{F}) = b(P')$ and $B(\widetilde{F}) = B(P')$. Finally, the third phase gives each robot its final destination by making the robots agree on a perfect matching between P' and $\widetilde{F}$, denoted by $M(P', \widetilde{F})$.

The following theorem has been shown in [10] that allows ψ_{PF} to concentrate each element of the $\gamma(P)$-decomposition of the current configuration P.

Theorem 11 [10] Let P be a set of points. Then P can be decomposed into subsets $\{P_1, P_2, \dots, P_m\}$ in such a way that each P_i is a vertex-transitive set of points regarding $\gamma(P)$. Furthermore, the robots can agree on a total ordering among the elements of the $\gamma(P)$-decomposition of P.

In [10], it is shown that there exists an ordering of the elements of $\{P_1, P_2, \dots, P_m\}$ so that P_1 is on $L(P)$, P_m is on $B(P)$, and P_{i+1} is not in the interior of $B(P_i)$. Additionally, robots can easily compute and agree on such ordering. In the following, we assume that $\{P_1, P_2, \dots, P_m\}$ is ordered in this way.

5.1 First phase: Symmetry breaking

In the first phase, robots execute a symmetry breaking algorithm ψ_{SYM} that translates an initial configuration P into another configuration P' that satisfies $\gamma(P') \in \varrho(P)$. Algorithm ψ_{SYM} is based on the "go-to-center" algorithm in [10] and it realizes symmetry breaking in an initial configuration P whose $\gamma(P)$-decomposition contains a regular tetrahedron, a regular octahedron, a cube, a regular dodecahedron, or a regular icosahedron. When robots form one of these polyhedra, they are on some rotation axes of $\gamma(P)$ and $\gamma(P) \notin \varrho(P)$. Algorithm ψ_{SYM} applies the "go-to-center" algorithm so that the robots on the rotation axes remove the rotation axes by leaving their current positions. Thus the symmetricity of the new configuration is a proper subgroup of the rotation group of the previous configuration. By repeating this procedure, the system reaches a configuration P' with $\gamma(P') \in \varrho(P)$. Additionally, ψ_{SYM} makes the robots repeat this procedure until no robot is on the rotation axes of the rotation group of the current configuration.

We first note that there is no way for the robots to reduce the rotation group of an initial configuration P when P forms a regular n-gon. Here, $\gamma(P) = D_n$ and $\varrho(P) = \{C_n, D_{n/2}\}$ if n is even, $\varrho(P) = \{C_n\}$ otherwise. Consider the case where n is even. To show the symmetricity, the robots either show an orientation of some single rotation axis or divide themselves into two groups to form $U_{D_{n/2},1}$. However, when $\sigma(P) = C_n$, the robots keep some regular n-gon forever. The robots neither show an agreement on the orientation of a single rotation axis nor divide themselves into two groups. Because the robots are oblivious, they do not remember the previous trials without recognizing $\sigma(P) = C_n$, and they keep on trying to show their symmetricity forever. We have the same situation when n is even. We avoid this infinite trial by leaving a regular n-gon as it is. Hence the proposed algorithm ψ_{SYM} do nothing when P forms a regular polygon. This is not a problem for Theorem 1 since from such P, the target pattern F satisfies $C_n, D_{n/2} \in \varrho(F)$ and hence $\gamma(F) \geq D_n$, and the robots do not need to break the symmetry.

Let $\{P_1, P_2, \ldots, P_m\}$ be the $\gamma(P)$-decomposition of an initial configuration P. We focus on the element that consists of points on rotation axes of $\gamma(P)$. In other words, we focus not on the coordinates of each point of P_i, but on the folding of P_i. We denote a polyhedron generated by a rotation group G and a seed point s with folding μ by $U_{G,\mu}$. For example, $U_{O,3}$ represents a regular cube.

We start with a symmetry breaking algorithm when the robots form one of the seven polyhedra. Hence we consider $U_{G,\mu}$ for $G \in \{T, O, I\}$ and $\mu > 1$. The proposed "go-to-center" algorithm is shown in Algorithm 1. If a current configuration forms one of the above seven polyhedra, Algorithm 1 makes each robot select an adjacent face and approach the center of the selected face, but stop ϵ before the center.

Lemma 12 Let P be an arbitrary initial configuration that forms a $U_{G,\mu}$ for $G \in \{T, O, I\}$ and $\mu > 1$. One step execution of Algorithm 1 translates P into another configuration P' that satisfies $\gamma(P') \in \varrho(P)$.

Proof. (Sketch.) Let P, P' be an initial configuration that forms one of the seven (semi-)regular polyhedra and a configuration obtained by one-step execution of Algorithm 1.

Algorithm 1 `Go-to-center`(P) for robot $r_i \in R$

Notation

P: The positions of robots forming an $U_{G,\mu}$ for $G \in \{T, O, I\}$ and $\mu > 1$ observed in Z_i.
p_i: Current position of r_i.
ϵ: an arbitrarily small distance compared to the distance between any two centers of the faces of P.

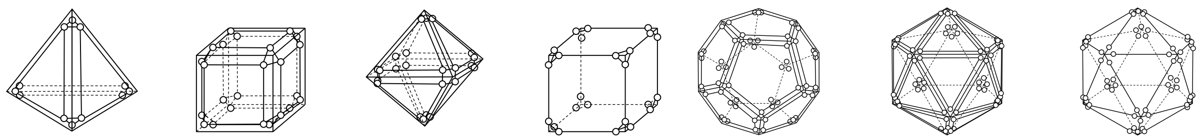
Algorithm

```
Switch (P) do
    Case P forms a cuboctahedron:
        Select an adjacent triangle face.
        Destination d is the point ε before the center of the selected
        face on the line from p_i to the center.
        Endcase
    Case P forms a icosidodecahedron:
        Select an adjacent pentagon face.
        Destination d is the point ε before the center of the selected
        face on the line from p_i to the center.
        Endcase
    Default:
        Select an adjacent face.
        Destination d is the point ε before the center of the selected
        face on the line from p_i to the center.
        Enddefault
Enddo
```

The proof follows the same idea as [10]. Let D be the set of all points that can be selected by the robots as their destinations in P. When P is a regular polyhedron, the points of D are placed around the vertices of the dual of P, which we call *base polyhedron* for D. For example, when P is a cube, the base polyhedron is a regular octahedron and the points of D are put around the corners of a regular octahedron (Figure 1(c)). When P is a cuboctahedron (an icosidodecahedron, respectively), the destinations are placed around the 3-fold rotation axes of O (the 5-fold rotation axes of I, respectively). In this case, we consider the cube (the regular icosahedron, respectively) as its base polyhedron.

Figure 1 shows the base polyhedron and D for each of the seven initial configurations. When P is a regular polyhedron, D forms a polyhedron obtained by cutting the vertices of the base polyhedron and bevel its original edges. For example, when P is a cube, D is a polyhedron obtained by a regular octahedron with above procedure and we call the polyhedron *ϵ-cantellated octahedron* (Figure 1(c)). In the same way, if P is a regular tetrahedron, D forms an *ϵ-cantellated tetrahedron* (Figure 1(a)), if P is a regular octahedron, D forms an *ϵ-cantellated cube* (Figure 1(b)), if P is a regular dodecahedron, D forms an *ϵ-cantellated icosahedron* (Figure 1(f)), and if P is a regular icosahedron, D forms an *ϵ-cantellated dodecahedron* (Figure 1(e)). On the other hand, when P is a semi-regular polyhedra, D forms a polyhedron obtained by cutting the vertices of the base polyhedron. For example, when P is a cuboctahedron, D is a polyhedron obtained from a cube by cutting its vertices, and we call the polyhedron an *ϵ-truncated cube* (Figure 1(d)). If P is a icosidodecahedron, D forms an *ϵ-truncated icosahedron* (Figure 1(g)).

Algorithm 1 makes the robots select a subset of size $|P|$ from



(a) For $U_{T,3}$ (b) For $U_{O,4}$ (c) For $U_{O,3}$ (d) For $U_{O,2}$ (e) For $U_{I,5}$ (f) For $U_{I,3}$ (g) For $U_{I,2}$

Fig. 1 Candidate set D for initial configuration $U_{G,\mu}$ for $G \in \{T, O, I\}$ and $\mu > 1$.

D. We can show that the symmetricity of any such subset has the rotation group that satisfies the statement. □

Now, we show the overview of algorithm ψ_{SYM} that translates an initial configuration P to another configuration P' that satisfies $\gamma(P') \in \varrho(P)$ if $\gamma(P) \notin \varrho(P)$. Algorithm ψ_{SYM} makes the robots on each type (i.e., fold) of the rotation axes leave the positions by repeating the following procedure: ψ_{SYM} first selects an element of the $\gamma(P)$-decomposition of the current configuration P that is on the rotation axes of $\gamma(P)$ and make the element shrink toward $b(P)$ so that the other robots keep the smallest enclosing ball. Then ψ_{SYM} makes the innermost robots execute the go-to-center algorithm (Algorithm 1) to eliminate the rotation axes. Hence, it takes at most two steps to eliminate one type of rotation axes. Because there are at most three types of rotation axes in any rotation group, the proposed algorithm terminates at most six steps.

Any terminal configuration P of ψ_{SYM} satisfies one of the following two properties:

(i) If $\gamma(P) \neq C_1$, then P is a regular n-gon or no robot is on the rotation axes of $\gamma(P)$.

(ii) $\gamma(P) = C_1$.

Hence P satisfies that the $\gamma(P)$-decomposition of P consists of elements of size $|\gamma(P)|$, which is shown to be useful in the pattern formation algorithm. Here, C_1-decomposition of P divides P into n subsets.

We can show that the movement of robots gradually eliminate the occupied rotation axes of $\gamma(P(0))$ without adding any new rotation axis. Thus we have Theorem 13. We omit the correctness proof of ψ_{SYM} because of the page restriction.

Theorem 13 Let P be an arbitrary initial configuration. Algorithm ψ translates P into another configuration P' that satisfies (i) $\gamma(P') \in \varrho(P)$ and (ii) all rotation axes of $\gamma(P')$ are unoccupied, in at most six steps.

5.2 Second phase: Agreement of $\widetilde{F}$

As we have already seen in Section 5.1, algorithm ψ_{SYM} translates an initial configuration P to another configuration P' that satisfies (i) $\gamma(P') \in \varrho(P)$, (ii) If $\gamma(P') \neq C_1$, then P' is a regular n-gon or no robot is on the rotation axes of $\gamma(P')$. Let $\{P'_1, P'_2, \dots, P'_m\}$ be the $\gamma(P')$-decomposition of P'. From the first property and the condition of Theorem 10, we have $\gamma(P') \in \varrho(F)$. From the second property, the size of each element $|P'_i|$ ($1 \leq i \leq m$) is $|\gamma(P')|$. The second property implies $\sigma(P') = \gamma(P')$ in the worst case and the robots forming each element may forever move symmetrically regarding $\gamma(P')$. Thus we make the robots forming each element move to symmetric positions.

Because $\gamma(P') \in \varrho(F)$, there exists an embedding of $\gamma(P')$ to unoccupied rotation axes of $\gamma(F)$. Thus when we fix F in P' in such a way, $\gamma(P')$-decomposition of F consists of elements of size $|\gamma(P')|$. We can overcome the symmetricity of each element of the $\gamma(P')$-decomposition of P' by assigning it to an element of $\gamma(P')$-decomposition of F. The proposed pattern formation algorithm ψ_{PF} first makes the robots agree on an embedding of F in P' so that $\gamma(P')$ overlaps unoccupied rotation axes of $\gamma(F)$. We denote the embedded target pattern by $\widetilde{F}$. The size of $\widetilde{F}$ is determined so that $b(P') = b(\widetilde{F})$ and $B(P') = B(\widetilde{F})$ hold. Then ψ_{PF} makes the robots compute a perfect matching between P' and $\widetilde{F}$ to assign a final destination to each robot.

However, $\gamma(P')$ is not sufficient to fix $\widetilde{F}$. For example, consider the case where P' is a pyramidal frustum with regular hexagon bases (thus $\gamma(P') = C_6$ and $\varrho(P') = \{C_6\}$) and F is an anti-prism with regular hexagon bases (thus $\gamma(F) = D_6$ and $\varrho(F) = \{D_3, C_6\}$). We can easily come up with an idea that we fix $\widetilde{F}$ considering the single rotation axis of $\gamma(P')$ as the principal axis of $\gamma(\widetilde{F})$. However, there are still infinite arrangement of $\widetilde{F}$ depending on the rotation of the hexagonal anti-prism. This is because the secondary axes of $\gamma(\widetilde{F})$ is not fixed. Algorithm ψ_{PF} fixes all rotation axes of $\widetilde{F}$ by using the positions of points of P'. For example, in this case, $\gamma(P')$-decomposition of P' consists of two regular hexagons and we select the first element as reference points so that the vertices of the regular hexagon fixes the three secondary axes of $\gamma(\widetilde{F})$. This also fixes the arrangement of the remaining three secondary axes of $\gamma(F)$ and $\widetilde{F}$. We call such reference points chosen from the elements of the $\gamma(P')$-decomposition of P' a *reference polygon*. The robots can agree on the reference polygon because of Theorem 11. Specifically, ψ_{PF} extracts reference polygon from P' and $\widetilde{F}$ and fixes $\widetilde{F}$ in P' by overlapping rotation axes of $\gamma(P')$ to unoccupied rotation axes of $\gamma(\widetilde{F})$ and the reference polygon of P' to that of $\widetilde{F}$.

5.3 Third phase: Matching $M(P', \widetilde{F})$

Let P' be a terminal configuration of ψ_{SYM} and $\widetilde{F}$ be the target pattern fixed in P'. As described in Section 5.2, the robots now compute a perfect matching between the points of P' and the points of $\widetilde{F}$ to finally form the target pattern.

We now consider the rotation group of $P' \cup \widetilde{F}$. We consider the rotations that matches the points of P' to P' and those of $\widetilde{F}$ to $\widetilde{F}$. Hence, $\gamma(P' \cup \widetilde{F}) \preceq \gamma(P')$. Actually, $\gamma(P' \cup \widetilde{F}) = \gamma(P')$ because $\gamma(P') \in \varrho(\widetilde{F})$ and each $G \in \varrho(\widetilde{F}) \preceq \gamma(F)$, i.e., any rotation of $\gamma(P')$ is applicable to $\widetilde{F}$. Hence, the group action of $\gamma(P')$ divides $P' \cup \widetilde{F}$ to a vertex-transitive set of points regarding $\gamma(P')$ so that each element consists of only the point of P' or only those of F. Additionally, each element consists of $|\gamma(P')|$ points since no robot is on $\gamma(P')$.

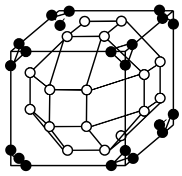


Fig. 2 The white circles are positions of the robots, and the black circles are the positions of destinations. P' forms a cantellated octahedron and $\widetilde{F}$ forms a truncated cube. Each robots has two nearest target points.

Algorithm 2 Pattern formation algorithm ψ_{PF} for robot $r_i \in R$

Notation

P: The positions of robots observed in Z_i.

p_i: current position of r_i.

Algorithm

If P is not a terminal configuration of ψ_{SYM} **then**

Execute ψ_{SYM}.

Else

Let $\widetilde{F}$ be the target pattern fixed in P.

Move to the matched point in $M(P, \widetilde{F})$.

Endif

Now, in the same way as [10], the robots can order the elements. Let $\{P'_1, P'_2, \ldots, P'_m\}$ and $\{\widetilde{F}_1, \widetilde{F}_2, \ldots, \widetilde{F}_m\}$ be the elements of P' and those of $\widetilde{F}$ that appears in the entire decomposition in this order. Then ψ_{PF} sends the robots forming P'_i to the positions of $\widetilde{F}_i$ for $1 \leq i \leq m$. In each element P'_i, each robot selects the nearest point in $\widetilde{F}_i$ as its destination. We first show that there exists a minimum weight perfect matching between the points of P'_i and $\widetilde{F}_i$, where the weight is the sum of distances between matched points.

Lemma 14 For each element P'_i and $\widetilde{F}_i$, there exists a minimum weight perfect matching between the points of P'_i and the points of $\widetilde{F}_i$.

Each point of $p \in P'_i$ may have multiple nearest destinations. Figure 2 shows an example where P'_i forms a cantellated cube and $\widetilde{F}_i$ forms a truncated cube. For each robot (white circle), there are two nearest destinations (black circles) around the corner of the cube. In such cases, we can show that the conflict forms a cycle around a rotation axis, and the robots can resolve it by a right-screw rule around the rotation axis with $b(P')$ being the positive direction.

We denote the entire matching obtained with these rules by $M(P', \widetilde{F})$. Remember that all computations consisting of finding reference polygon, fixing $\widetilde{F}$, decomposition $P' \cup \widetilde{F}$, and computing $M(P', \widetilde{F})$ is done in one Compute phase in a terminal configuration of ψ_{SYM}. Finally, robots move the corresponding position in $M(P', \widetilde{F})$ to complete the pattern formation.

We finally show the proposed pattern formation algorithm ψ_{PF} in Algorithm 2.

6. Discussion on Theorem 2

In [10], the following necessary and sufficient condition for the plane formation problem has been shown: The oblivious FSYNC robots cannot form a plane from an initial configuration P if and only if $\gamma(P)$ is 3D and the size of each element of the $\gamma(P)$-decomposition $\{P_1, P_2, \ldots, P_m\}$ of P is in $\{12, 24, 60\}$. Remember that $\{12, 24, 60\}$ are the order of the 3D rotation groups, T, O, and I. Hence when the condition is satisfied, $\varrho(P)$ contains at least one 3D rotation group. From Lemma 8, oblivious FSYNC robots cannot form a plane when $\varrho(P)$ contains a 3D rotation group. In [10], the authors showed a plane formation algorithm that uses the "go-to-center" algorithm when $\gamma(P)$-decomposition of P contains a regular tetrahedron, a cube, a regular octahedron, a regular dodecahedron, or an icosidodecahedron. Hence, we can rephrase the necessary and sufficient condition of [10] as shown in Theorem 2.

7. Conclusion

We have shown a necessary and sufficient condition for the oblivious FSYNC robots to form a given target pattern. We introduce the notion of symmetricity of positions of robots in 3D-space and used it to characterize the pattern formation problem. The necessary and sufficient condition is a generalization of that in 2D-space. (See [7], [8], [9].) The results of this paper are directly extended to target patterns with multiplicities and non-oblivious robots. We further rephrase the existing necessary and sufficient condition for the plane formation problem [10]. Our future direction is to consider formation problems with weaker assumptions, such as asynchrony among robots, non-rigid movement, and limited visibility.

References

[1] H.S.M. Coxeter, Regular polytopes, Dover Publications, 1973.

[2] P. Cromwell, Polyhedra, University Press, 1997.

[3] M. Cieliebak, P. Flocchini, G. Prencipe, and N. Santoro, Solving the robots gathering problem, *Proc. of the 30th International Colloquium on Automata, Languages and Programming,* pp.1181–1196, 2003.

[4] Y. Dieudonné, F. Petit, and V. Villain, Leader election problem versus pattern formation problem. *Proc. of the 24th International Symposium on Distributed Computing,* pp.267–281, 2010.

[5] P. Flocchini, G. Prencipe, N. Santoro, and P. Widmayer, Arbitrary pattern formation by asynchronous, anonymous, oblivious robots, *Theoretical Computer Science,* **407**, pp.412–447, 2008.

[6] P. Flocchini, G. Prencipe, and N. Santoro, Distributed Computing by Oblivious Mobile Robots, Synthesis Lectures on Distributed Computing Theory 10, Morgan & Claypool Publishers, 2012.

[7] N. Fujinaga, Y. Yamauchi, H. Ono, S. Kijima, and M. Yamashita, Pattern Formation by Oblivious Asynchronous Mobile Robots, *SIAM Journal on Computing,* **44**, 3, pp.740–785, 2015.

[8] I. Suzuki and M. Yamashita, Distributed anonymous mobile robots: Formation of geometric patterns, *SIAM Journal on Computing,* **28**, 4, pp.1347–1363, 1999.

[9] M. Yamashita and I. Suzuki, Characterizing geometric patterns formable by oblivious anonymous mobile robots, *Theoretical Computer Science*, **411**, pp.2433–2453, 2010.

[10] Y. Yamauchi, T. Uehara, S. Kijima, and M. Yamashita, Plane Formation by Synchronous Mobile Robots in the Three Dimensional Euclidean Space, *Proc. of the 29th International Symposium on Distributed Computing (DISC2015),* pp.92-106, 2015