Compatibility of Convergence Algorithms for Autonomous Mobile Robots (Extended Abstract)*

Yuichi Asahiro[1]** and Masafumi Yamashita[2]

[1] Kyushu Sangyo University, Fukuoka, Japan
asahiro@is.kyusan-u.ac.jp
[2] Professor Emeritus, Kyushu University, Fukuoka, Japan
masafumi.yamashita@gmail.com

Abstract. We investigate autonomous mobile robots in the Euclidean plane. A robot has a function called *target function* to decides the destination from the robots' positions, and operates in Look-Compute-Move cycles, i.e., identifies the robots' positions, computes the destination by the target function, and then moves there. Robots can have different target functions. Let Φ and Π be a set of target functions and a problem, respectively. If the robots whose target functions are chosen from Φ always solve Π, we say that Φ is *compatible* with respect to Π. Suppose that Φ is compatible with respect to Π. Then two swarms controlled by (possibly different) target functions in Φ can merge to form a larger swarm, and a broken robot can be replaced with another robot with any target function in Φ, keeping the correctness of solving Π. We investigate the convergence, the gathering, and some fault tolerant convergence problems, assuming crash failures, from the view point of compatibility.

Keywords: Autonomous mobile robot · Compatibility · Convergence · Crash fault · Gathering.

1 Introduction

Over the last three decades, swarms of autonomous mobile robots have obtained much attention in a variety of contexts. Among them is understanding solvable problems by a swarm consisting of many simple and identical robots in a distributed manner, which has been constantly attracting researchers in distributed computing society [1–3, 5, 6, 8–20].

Many of the works mentioned above adopt the following robot model. The robots look identical and indistinguishable. Each robot is represented by a point that moves in the Euclidean plane. It lacks identifier and communication devices, and operates in Look-Compute-Move cycles. When a robot starts a cycle,

* Due to the space limitation, we omit most of the proofs and some contributions. The full version of the paper [4] contains them.

** Corresponding author.

it identifies the multiset of the robots' positions in its local x-y coordinate system such that it is right-handed, and its origin is always the position of the robot, computes the destination point using a target function[3] based only on the multiset identified, and then moves towards the target position. If each cycle starts at a time t and finishes, reaching the target position, before (not including) $t+1$, for some integer t, the scheduler is said to be *semi-synchronous* ($\mathcal{SSYNC}$). If cycles can start and end any time (even on the way to the target point), it is *asynchronous* ($\mathcal{ASYNC}$).

This paper investigates several *convergence problems*, e.g., [2, 8–10, 13, 14, 16, 18]. The simplest convergence problem requires the robots to converge to a single point. For the $\mathcal{SSYNC}$ model, the problem is solvable for robots with unlimited visibility [18], and is also solvable for robots with limited visibility [2].

Under the $\mathcal{ASYNC}$ model, it is solvable by a target function called CoG, which always outputs the center of gravity of the robots' positions [8]. In [8], the authors also showed that CoG correctly works under the sudden-stop model, under which the movement of a robot towards the center of gravity might stop on the way after traversing at least some fixed distance. This implies that the robots can correctly converge to a point, even when they are controlled by different target functions as long as they always move robots towards the current center of gravity over distance at least some fixed constant. This idea is extended in [10]: The authors proposed the *δ-inner* property[4] of target functions, and showed that the robot system converges to a point if all robots take δ-inner target functions, provided $\delta \in (0, 1/2]$. Finally [16] gives a convergence algorithm for robots with limited visibility under the $\mathcal{ASYNC}$ model.

Consider a problem Π and a set of target functions Φ. If the robots whose target functions are chosen from Φ always solve Π, we say that Φ is *compatible with respect to* Π. For example, every (non-empty) set of $(1/2)$-inner functions is compatible with respect to the convergence problem [10].

If a singleton $\{\phi\}$ is compatible with respect to Π, we abuse to say that target function ϕ is an *algorithm*[5] for Π. If a set Φ of target functions is compatible with respect to Π, every target function $\phi \in \Phi$ is an algorithm for Π. (The converse is not always true.) Thus there is an algorithm for Π, if and only if there is a compatible set Φ with respect to Π. We say that a problem Π is *solvable*, if there is a compatible set Φ with respect to Π, meaning that there is an algorithm for Π.

We would like to find a large compatible set Φ with respect to Π. That Π has a large compatible set with respect to Π implies that Π has many algorithms.

---

[3] Roughly, a target function is a function from $(R^2)^n$ to R^2, where R is the set of real numbers and n is the number of robots, i.e., given a snapshot in $(R^2)^n$, it returns a destination point in R^2. Later, we define a target function a bit more carefully.

[4] Let P, D, and $\boldsymbol{o}$ be the multiset of robots' positions, the axes aligned minimum box containing P, and its center, respectively. Define $\delta * D = \{\delta \boldsymbol{x} + (1-\delta)\boldsymbol{o} : \boldsymbol{x} \in D\}$. A function ϕ is δ-inner, if $\phi(P)$ is included in $\delta * D$ for any P.

[5] Here, we abuse term "algorithm," since an algorithm must have a finite description. A target function may not. To compensate the abuse, we insist on giving a finite procedure when we show the existence of a target function.

The difficulty of problems might be compared by the sizes of their compatible sets. A problem Π with a large compatible set Φ seems to have some practical merits. Two swarms both of which are controlled by target functions in Φ (which may be produced by different makers) can merge to form a larger swarm, keeping the correctness of solving Π. When a robot breaks down, we can safely replace it with another robot, as long as it is controlled by a target function in Φ.

Fault Tolerant Convergence Problems. This paper investigates three fault-tolerant convergence problems, besides the convergence and the gathering problems. We consider only crash faults: A faulty robot can stop functioning at any time, becoming permanently inactive. A faulty robot may not cause a malfunction, forever. We cannot distinguish such a robot from non-faulty ones. Let n and $f(\leq n-1)$ be the number of robots and the number of faulty robots.

The *fault-tolerant (n,f)-convergence problem* (FC(f)) is the problem to find an algorithm which ensures that, as long as at most f robots are faulty, *all non-faulty robots* converge to a single point. The *fault-tolerant (n,f)-convergence problem to f points* (FC(f)-PO) is the problem to find an algorithm which ensures that, as long as at most f robots are faulty, *all robots* (including faulty ones) converge to at most f points. All non-faulty robots need not converge to the same point. If f faulty robots have crashed at different positions, each non-faulty robot must converge to one of the faulty robots. The *fault-tolerant (n,f)-convergence problem to a convex f-gon* (FC(f)-CP) is the problem to find an algorithm which ensures that, as long as at most f robots are faulty, the convex hull of the positions of *all robots* (including faulty ones) converges to a convex h-gon CH for some $h \leq f$, in such a way that, for each vertex of CH, there is a robot that converges to the vertex.

Since an algorithm for FC(1)-PO solves FC(1), the former is not easier than the latter. (Note that for $f \geq 2$, an algorithm for FC(f)-PO may not solve FC(f).) Since an algorithm for FC(f)-PO solves FC(f)-CP, again the former is not easier than the latter. In [8], the authors showed that, for all $f \leq n-2$, CoG is an algorithm for FC(f) under the $\mathcal{ASYNC}$ model. As far as we know, FC(f)-PO and FC(f)-CP have not been investigated so far.

Gathering Problem. The *gathering problem* requires the robots to gather in the exactly the same location. For $\mathcal{SSYNC}$, the gathering problem is not solvable if $n = 2$. If $n > 2$, it is solvable, provided that all robots initially occupy distinct positions [18]. For $\mathcal{ASYNC}$, the same results hold [7]. The gathering problem has been investigated under a variety of assumptions [1, 5, 7, 11–14].

Contributions. Let R be the set of real numbers. Formally, a *target function* ϕ is a function from $(R^2)^n$ to $R^2 \cup \{\perp\}$ for all $n \geq 1$ such that $\phi(P) = \perp$ if and only if $(0,0) \notin P$. Here, $\perp$ is a special symbol to denote that $(0,0) \notin P$. Suppose that a robot r identifies a multiset P of n points, which are the positions of the robots in its local x-y coordinate system Z, in Look phase. Then $(0,0) \in P$.[6]

[6] That $(0,0) \notin P$ means an error of eye sensor, which we assume will not occur, in this paper.

Table 1. The compatibility of a set Φ of target functions with respect to a problem Π, taking its scale $\alpha(\Phi)$ as a parameter. Each entry contains the status A, E, N, or ? of the compatibility of Φ with respect to Π (and the theorem/corollary/observation/citation number establishing the result in parentheses). Letter 'A' means that every Φ such that $\alpha(\Phi)$ is in the range is compatible with respect to Π. Letter 'N' means that any Φ such that $\alpha(\Phi)$ is in the range is not compatible with respect to Π, which indicates the absence of an algorithm. Letter 'E' means that some Φ is compatible, while some other is not, which indicates the existence of an algorithm. Letter '?' means that the answer is unknown.

problem Π	scale $\alpha(\Phi)$		
	$\alpha(\Phi) = 0$	$0 < \alpha(\Phi) < 1$	$\alpha(\Phi) = 1$
Convergence	A (Thm. 1 [8])	A (Thm. 2)	E (Thm. 3)
FC(1)	A ([8])	A (Cor. 1)	E (Thm. 5)
FC(1)-PO	A (Thm. 4)	A (Thm. 4)	E (Thm. 5)
FC(f) ($f \geq 2$)	A (Thm. 6 [8])	E (Thm. 7)	E (Cor. 2)
FC(f)-CP ($f \geq 2$)	A (Thm. 8)	A (Thm. 8)	E (Trivial)
FC(2)-PO	N (Thm. 9)	N (Thm. 9)	E (Thm. 10)
FC(f)-PO ($f \geq 3$)	N (Thm. 9)	N (Thm. 9)	?
Gathering	N (Thm. 12)	N (Thm. 12)	E (Thm. 11 [18])

Using its target function ϕ, r computes the target point $\boldsymbol{x} = \phi(P)$ in Compute phase. Then it moves to $\boldsymbol{x}$ ($\neq \perp$) in Z in Move phase.

Let the convex hull and the center of gravity of P be $CH(P)$ and $g(P)$, respectively. For any $0 \leq d$, let $d * CH(P) = \{d\boldsymbol{x} + (1-d)g(P) : \boldsymbol{x} \in CH(P)\}$. The *scale* $\alpha(\phi)$ of a target function ϕ is defined by

$$\alpha(\phi) = \sup_{P \in (R^2)^n} \alpha(\phi, P),$$

where $\alpha(\phi, P)$ is the smallest d satisfying $\phi(P) \in d * (CH(P))$.[7] Then the scale of a set Φ of target functions is defined by

$$\alpha(\Phi) = \sup_{\phi \in \Phi} \alpha(\phi).$$

The only target function ϕ satisfying $\alpha(\phi) = 0$ is CoG. Thus the set Φ of target functions satisfying $\alpha(\Phi) = 0$ is a singleton {CoG}. The idea of scale is similar to that of the δ-inner property in [10], and more directly embodies the idea behind the δ-inner target function.

Our contributions are summarized in Table 1. For example, the entry of Convergence and $\alpha(\phi) = 0$ is A. Thus {CoG} is compatible with respect to the convergence problem, or CoG is an algorithm for the convergence problem, as [8] shows. Not only the case $\alpha(\phi) = 0$, but also the case $0 < \alpha(\Phi) < 1$, every Φ is compatible with respect to the convergence problem.

Organization. After introducing the robot model in Section 2, we investigate the convergence problem in Section 3. In Section 4, we discuss the compatibilities

[7] For the sake of completeness, we assume that $\alpha(\phi, P) = 0$ when $\phi(P) = \perp$.

of two convergence problems FC(1) and FC(1)-PO. Sections 5 and 6 respectively investigate the compatibilities of FC(f) and FC(f)-CP for $f \geq 2$. Section 7 first shows that a target function ϕ is an algorithm for FC(f)-PO for $f \geq 2$, only if $\alpha(\phi) \geq 1$. We then presents an algorithm $\psi_{(n,2)}$ for FC(2)-PO. Section 8 investigates the gathering problem to show the difference between this and the convergence problems. We conclude the paper in Section 9.

2 The Model

Consider a robot system $\mathcal{R}$ consisting of n robots $r_1, r_2, \ldots, r_n$. Each robot r_i has its own unit of length, and a local compass defining an local x-y coordinate system Z_i, which is assumed to be right-handed and self-centric, i.e., its origin $(0,0)$ is always the position of r_i. We also assume that r_i has the strong multiplicity detection capability, i.e., it can count the number of robots resides at a point. Given a target function ϕ_i, each robot $r_i \in \mathcal{R}$ repeatedly executes a Look-Compute-Move cycle:

Look: Robot r_i identifies the multiset P of the robots' positions (including the one of r_i) in Z_i. Since r_i has the strong multiplicity detection capability, it can identify P not only distinct positions of P.

Compute: Robot r_i computes $\boldsymbol{x}_i = \phi_i(P)$. (We do not mind even if ϕ_i is not computable. We simply assume that $\phi_i(P)$ is given by an oracle.)

Move: Robot r_i moves to $\boldsymbol{x}_i$. We assume that r_i always reaches $\boldsymbol{x}_i$ before this Move phase ends.

We assume a discrete time $0, 1, \ldots$. At each time $t \geq 0$, the scheduler activates some unpredictable subset (that may be none or all) of robots. Then activated robots execute a cycle which starts at t and ends before (not including) $t+1$ (unless it has crashed), i.e., the scheduler is semi-synchronous ($\mathcal{SSYNC}$). Let Z_0 be the global x-y coordinate system, which is right-handed and is not accessible by any robot r_i. The coordinate transformation from Z_i to Z_0 is denoted by γ_i. We use Z_0 and γ_i just for the purpose of explanation.

The position of robot r_i at time t in Z_0 is denoted by $\boldsymbol{x}_t(r_i)$. Then $P_t = \{\boldsymbol{x}_t(r_i) : 1 \leq i \leq n\}$ is a multiset representing the positions of all robots at time t, and is called the *configuration* at t.

Given an initial configuration P_0, an assignment $\mathcal{A}$ of a target function ϕ_i to each robot r_i, and an $\mathcal{SSYNC}$ activation schedule, the execution of $\mathcal{R}$ is a sequence $\mathcal{E} : P_0, P_1, \ldots, P_t, \ldots$ of configurations starting from P_0. Here, for all r_i and $t \geq 0$, if r_i is not activated at t, $\boldsymbol{x}_{t+1}(r_i) = \boldsymbol{x}_t(r_i)$. Otherwise, if it is activated, r_i identifies $Q_t^{(i)} = \gamma_i^{-1}(P_t)$ in Z_i, computes $\boldsymbol{y} = \phi_i(Q_t^{(i)})$, and moves to $\boldsymbol{y}$ in Z_i.[8] Then $\boldsymbol{x}_{t+1}(r_i) = \gamma_i(\boldsymbol{y})$. We assume that the scheduler is fair: It activates every robot infinitely many times. Throughout the paper, we regard the scheduler as an adversary.

[8] Since $(0,0) \in Q_t^{(i)}$ by definition, $\boldsymbol{y} \neq \perp$.

We introduce several notations. Let $P \in (R^2)^n$. The distinct points of P is denoted by $\overline{P}$. Then $|P|$ (resp. $|\overline{P}|$) denotes the number of points (resp. the number of distinct points) in P. Let $CH(P)$ be the convex hull of P. We sometimes denote $CH(P)$ by a sequence of vertices of $CH(P)$ appearing on the boundary counter-clockwise. The center of gravity $g(P)$ of P is defined by $g(P) = \sum_{\boldsymbol{x} \in P} \boldsymbol{x}/n$. For two points $\boldsymbol{x}$ and $\boldsymbol{y}$ in R^2, $dist(\boldsymbol{x}, \boldsymbol{y})$ denotes the Euclidean distance between $\boldsymbol{x}$ and $\boldsymbol{y}$. For a set $B(\subseteq R^2)$ of points and a point $\boldsymbol{a} \in R^2$, $dist(\boldsymbol{a}, B) = \min_{\boldsymbol{x} \in B} dist(\boldsymbol{a}, \boldsymbol{x})$. Finally, let $\mathcal{P} = \{P \in (R^2)^n : (0,0) \in P,\ n \geq 1\}$. We regard $\mathcal{P}$ as the domain of target functions.

3 Convergence Problem

We investigate the convergence problem, provided that all robots are non-faulty. For any $0 \leq \alpha \leq 1$, consider a target function CoG_α defined by $\mathrm{CoG}_\alpha(P) = (1-\alpha)g(P)$, for any $P \in \mathcal{P}$. The scale of CoG_α is α, and $\mathrm{CoG}_0 = \mathrm{CoG}$. The following theorem holds, since CoG works correctly under the sudden-stop model.

Theorem 1 ([8]). *For any $0 \leq \alpha < 1$, let $\Phi_\alpha = \{\mathrm{CoG}_\alpha\}$. Then Φ_α is compatible with respect to the convergence problem, or equivalently, CoG_α is an algorithm for the convergence problem.*

We extend Theorem 1 to have the following theorem.

Theorem 2. *Let Φ be a set of target functions such that $0 \leq \alpha(\Phi) < 1$. Then Φ is compatible with respect to the convergence problem.*

Proof. (Sketch) Let $\phi_i \in \Phi$ be the target function taken by robot r_i for $i = 1, 2, \ldots, n$. Let $\alpha(\phi_i) = \alpha_i$ and $\alpha = \max_{1 \leq i \leq n} \alpha_i$. Then $\alpha \leq \alpha(\Phi) < 1$. Consider any execution $\mathcal{E} : P_0, P_1, \ldots$ starting from any initial configuration P_0. We show that P_t converges to a point.

Suppose that $P_t = \{\boldsymbol{x}, \boldsymbol{x}, \ldots, \boldsymbol{x}\}$ at some time t, i.e., $|\overline{P_t}| = 1$. Since $\boldsymbol{g}_t = g(P_t) = \boldsymbol{x}$, $P_{t+1} = P_t$. Thus convergence has already been achieved. We assume without loss of generality that $|\overline{P_t}| \geq 2$ for all $t \geq 0$.

Let $A_t \subseteq \mathcal{R}$ be the set of robots activated at time t. If $\boldsymbol{x}_t(r) = \boldsymbol{g}_t$ for all $r \in A_t$, $P_{t+1} = P_t$ holds. However, there is a robot r such that $\boldsymbol{x}_t(r) \neq \boldsymbol{g}_t$ since $|\overline{P_t}| \geq 2$, and r is eventually activated by the fairness of scheduler. Thus, without loss of generality, we assume that there is a robot $r \in A_t$ such that $\boldsymbol{x}_t(r) \neq \boldsymbol{g}_t$, and that $P_{t+1} \neq P_t$ holds for all $t \geq 0$.

We denote $CH(P_t)$ by CH_t. Since $\alpha < 1$, $CH_{t+1} \subseteq CH_t$, which implies that CH_t converges to a convex k-gon CH (including a point and a line segment) for some positive integer k. We show that CH is a point, i.e., $k = 1$.

Let $\boldsymbol{p}_0, \boldsymbol{p}_1, \ldots, \boldsymbol{p}_{k-1}$ be the vertices of CH aligned counter-clockwise on the boundary. To derive a contradiction, we assume that $k \geq 2$. For any pair (i, j) $(0 \leq i < j \leq k-1)$, let $L_{(i,j)} = dist(\boldsymbol{p}_i, \boldsymbol{p}_j)$, and $L = \min_{0 \leq i < j \leq k-1} L_{(i,j)}$. Since CH_t converges to CH, for any $0 < \epsilon \ll (1-\alpha)L/n$, there is a time instant t_0 such that, for all $t \geq t_0$, $CH \subseteq CH_t \subseteq N_\epsilon(CH)$. For any vertex $\boldsymbol{p}$ of CH, $dist(\boldsymbol{p}, \alpha * CH_t) > (1-\alpha)(L/n - \epsilon) - \epsilon \gg \epsilon$, since $dist(\boldsymbol{p}, \boldsymbol{g}_t) > L/n - \epsilon$.

Suppose that a robot r is activated at some time $t \geq t_0$. Then $\boldsymbol{x}_{t+1}(r) \in \alpha * CH_t$, which implies that $\boldsymbol{x}_{t+1}(r) \notin N_\epsilon(\boldsymbol{p})$, for any vertex $\boldsymbol{p}$ of CH. If r is reactivated at some time $t' > t$ for the first time after t, since $CH_{t'} \subseteq CH_t$ and $\boldsymbol{x}_{t'+1}(r) \in \alpha * CH_{t'}$, $\boldsymbol{x}_{t'+1}(r) \notin N_\epsilon(\boldsymbol{p})$, for any vertex $\boldsymbol{p}$ of CH. Therefore, for any $t' > t$ and any vertex $\boldsymbol{p}$ of CH, $\boldsymbol{x}_{t'}(r) \notin N_\epsilon(\boldsymbol{p})$.

On the other hand, all robots will be activated infinitely many times after time t, by the fairness of scheduler. It is a contradiction to the assumption that CH_t converges to CH, since there is a time instant $t' > t$ such that for any robot r and any vertex $\boldsymbol{p}$ of CH, $\boldsymbol{x}_{t'}(r) \notin N_\epsilon(\boldsymbol{p})$ holds. □

Let Φ and Φ' be two sets of target functions. If $\alpha(\Phi) < 1$ and $\alpha(\Phi') < 1$, Φ, Φ', and $\Phi \cup \Phi'$ are all compatible with respect to the convergence problem by Theorem 2. However, the following claim does not hold:

> If both of Φ and Φ' are compatible with respect to the convergence problem, so is $\Phi \cup \Phi'$.

To observe this fact, examine two target functions ϕ_T and ϕ_S. For a configuration P, define a condition Ψ as follows:

Ψ: $|P| = 7$, $(0,0) \in P$, $P = T \cup S$, T is an equilateral triangle, S is a square, T and S have the same side length, and T and S do not overlap.

[Target function ϕ_T]

1. If P satisfies Ψ:
   - **(a)** If $(0,0) \in T$, $\phi_T(P)$ is the middle point on the line segment connecting $(0,0)$ and $g(T)$.
   - **(b)** If $(0,0) \in S$, $\phi_T(P) = g(P)$.
2. If P does not satisfy Ψ: $\phi_T(P) = g(P)$.

[Target function ϕ_S]

1. If P satisfies Ψ:
   - **(a)** If $(0,0) \in S$, $\phi_S(P)$ is the middle point on the line segment connecting $(0,0)$ and $g(S)$.
   - **(b)** If $(0,0) \in T$, $\phi_S(P) = g(P)$.
2. If P does not satisfy Ψ: $\phi_S(P) = g(P)$.

Recall that $g(P)$, $g(T)$, and $g(S)$ are the centers of gravity of P, T, and S, respectively, and that when a robot identifies P in Look phase, $(0,0)$ always in P, which corresponds to its current position.

Let us observe that $\alpha(\phi_T) = 1$. Since $\phi_T(P) \in CH(P)$ for all P, $\alpha(\phi_T) \leq 1$. To see that $\alpha(\phi_T) \geq 1$, consider any number $0 < a < 1$. It is easy to construct a P satisfying Ψ such that $\frac{dist((0,0),g(T))}{dist((0,0)),g(P))} < a$, which implies that $\alpha(\phi_T) > 1 - a$. Thus $\alpha(\phi_T) = 1$ by the definition of α. By the same argument, $\alpha(\phi_S) = 1$.

Theorem 3. *Both $\Phi_T = \{\phi_T\}$ and $\Phi_S = \{\phi_S\}$ are compatible with respect to the convergence problem, but $\Phi = \Phi_T \cup \Phi_S$ is not.*

4 Convergence When at Most One Robot Crashes

We investigate the fault-tolerant $(n, 1)$-convergence problem (FC(1)) and the fault-tolerant $(n, 1)$-convergence problem to a point (FC(1)-PO). There is an algorithm for FC(1) [8], but FC(1)-PO is not easier than FC(1). We have the following theorem, which implies the existence of an algorithm for FC(1)-PO.

Theorem 4. *Let Φ be a set of target functions such that $0 \leq \alpha(\Phi) < 1$. Then Φ is compatible with respect to FC(1)-PO.*

Corollary 1. *Let Φ be a set of target functions such that $0 \leq \alpha(\Phi) < 1$. Then Φ is compatible with respect to FC(1).*

Next we reconsider the target functions ϕ_T and ϕ_S.

Theorem 5. *Both $\Phi_T = \{\phi_T\}$ and $\Phi_S = \{\phi_S\}$ are compatible with respect to FC(1)-PO. However, $\Phi = \Phi_T \cup \Phi_S$ is not. Recall that $\alpha(\Phi_T) = \alpha(\Phi_S) = \alpha(\Phi) = 1$.*

5 FC(f) for $f \geq 2$

We go on the fault tolerant (n, f)-convergence problem (FC(f)) for $f \geq 2$. Since CoG is an algorithm for FC(f) [8], the next theorem holds.

Theorem 6 ([8]). *Suppose that $f \leq n - 1$. The set $\Phi_0 = \{\text{CoG}\}$ is compatible with respect to FC(f), or equivalently, a set Φ of target functions is compatible with respect to FC(f), if $\alpha(\Phi) = 0$.*

Corollary 1 states that every set Φ of target functions such that $0 \leq \alpha(\Phi) < 1$ is compatible with respect to FC(1). In contrast, for any $2 \leq f \leq n - 1$ and $0 < \alpha < 1$, there is a set Φ of two target functions such that (1) $\alpha(\Phi) = \alpha$, (2) each target function in Φ is compatible with respect to FC(f), but (3) Φ is not compatible with respect to FC(f). We use target functions $\xi_{(\alpha,n)}$ and $\xi'_{(\alpha,n)}$. Let $\ell = \lfloor \frac{n-2}{2} \rfloor$ and $\ell' = \lceil \frac{n-2}{2} \rceil$. Thus $\ell + \ell' = n - 2$. For a configuration P, define a condition Ψ^+ by a conjunction of two conditions (i) and (ii).

Ψ^+: **(i)** $P = \{\boldsymbol{p}_1, \boldsymbol{p}_2, \ldots, \boldsymbol{p}_n\} \subseteq \overline{\boldsymbol{p}_1\boldsymbol{p}_n}$, where $\boldsymbol{p}_1, \boldsymbol{p}_{\ell+1}, \boldsymbol{p}_{\ell+2}, \boldsymbol{p}_{\ell+3}$ are distinct and aligned on $\overline{\boldsymbol{p}_1\boldsymbol{p}_n}$ in this order, $\boldsymbol{p}_1 = \boldsymbol{p}_2 = \cdots = \boldsymbol{p}_\ell$, i.e., the multiplicity of $\boldsymbol{p}_1$ is ℓ, $\boldsymbol{p}_{\ell+3} = \boldsymbol{p}_{\ell+4} = \cdots = \boldsymbol{p}_n$, i.e., the multiplicity of $\boldsymbol{p}_{\ell+3}$ is ℓ'.

(ii) Let $L = dist(\boldsymbol{p}_1, \boldsymbol{p}_n)$. Then $dist(\boldsymbol{p}_1, \boldsymbol{p}_{\ell+1}) = \frac{1}{2}L$. If n is even, $dist(\boldsymbol{p}_{\ell+1}, \boldsymbol{p}_{\ell+2}) = \frac{\alpha n}{2(\alpha+n-1)}L$; otherwise, if it is odd, $dist(\boldsymbol{p}_{\ell+1}, \boldsymbol{p}_{\ell+2}) = \frac{(2\alpha-1)n+(1-\alpha)}{2(\alpha(n+1)-1)}L$.

[Target function $\xi_{(\alpha,n)}$]

1. If P satisfies Ψ^+, $\xi_{(\alpha,n)}(P)$ is
   (a) $\boldsymbol{p}_{\ell+2}$, if $\boldsymbol{p}_{\ell+1} = (0, 0)$,

(b) $(0,0)$, if $\boldsymbol{p}_{\ell+2} = (0,0)$, and
(c) $g(P)$, otherwise.
2. If P does not satisfy Ψ^+, $\xi_{(\alpha,n)}(P) = g(P)$.

[Target function $\xi'_{(\alpha,n)}$]
1. If P satisfies Ψ^+, $\xi'_{(\alpha,n)}(P)$ is
(a) $(0,0)$, if $\boldsymbol{p}_{\ell+1} = (0,0)$,
(b) $\alpha\boldsymbol{p}_1 + (1-\alpha)g(P)$, if $\boldsymbol{p}_{\ell+2} = (0,0)$, and
(c) $g(P)$, otherwise.
2. If P does not satisfy Ψ^+, $\xi'_{(\alpha,n)}(P) = g(P)$.

Theorem 7. *For any $2 \leq f \leq n-1$ and $0 < \alpha < 1$, (1) $\alpha(\xi_{(\alpha,n)}) = \alpha(\xi'_{(\alpha,n)}) = \alpha$, (2) both of $\Phi = \{\xi_{(\alpha,n)}\}$ and $\Phi' = \{\xi'_{(\alpha,n)}\}$ are compatible with respect to FC(f), but (3) $\Phi \cup \Phi'$ is not.*

Before closing this section, we examine the case $\alpha = 1$.

Corollary 2. *For any $2 \leq f \leq n-1$, there are two target functions $\xi_{(1,n)}$ and $\xi'_{(1,n)}$ such that (1) $\alpha(\xi_{(1,n)}) = \alpha(\xi'_{(1,n)}) = 1$, (2) both of $\Phi = \{\xi_{(1,n)}\}$ and $\Phi' = \{\xi'_{(1,n)}\}$ are compatible with respect to FC(f), but (3) $\Phi \cup \Phi'$ is not.*

6 FC(f)-CP for $f \geq 2$

We next investigate the fault tolerant (n,f)-convergence problem to a convex f-gon (FC(f)-CP). FC(1)-CP is FC(1)-PO. FC(f)-CP seems to be substantially easier than FC(f) (and FC(f)-PO), since the convergence of CH_t to a convex f-gon does not always mean the convergence of P_t. We have the following theorem.

Theorem 8. *Let Φ be any set of target functions such that $0 \leq \alpha(\Phi) < 1$. Then Φ is compatible with respect to FC(f)-CP for any $2 \leq f \leq n-1$.*

Let Φ and Φ' be any sets of target functions such that $\alpha(\Phi) < 1$ and $\alpha(\Phi') < 1$ hold. Then all of Φ, Φ', and $\Phi \cup \Phi'$ are compatible with respect to FC(f)-CP for all $2 \leq f \leq n-1$, since $\alpha(\Phi \cup \Phi') < 1$, by Theorem 8. However, we cannot extend this observation to include the case $\alpha = 1$. Consider the following two target functions τ and τ' for three robots.

[Target function τ]
1. If $P = \{\boldsymbol{p}_1, \boldsymbol{p}_2, \boldsymbol{p}_3\}$ is a triangle such that $\angle\boldsymbol{p}_1 < \angle\boldsymbol{p}_2 < \angle\boldsymbol{p}_3$, where $\angle\boldsymbol{p}_i$ is the angle of vertex $\boldsymbol{p}_i$ of the triangle, $\tau(P)$ is
(a) $g(P)$ if $\boldsymbol{p}_1 = (0,0)$, and
(b) $\boldsymbol{p}_1$, otherwise.
2. Otherwise, $\tau(P) = g(P)$.

[Target function τ']
1. If $P = \{\boldsymbol{p}_1, \boldsymbol{p}_2, \boldsymbol{p}_3\}$ is a triangle such that $\angle\boldsymbol{p}_1 < \angle\boldsymbol{p}_2 < \angle\boldsymbol{p}_3$, where $\angle\boldsymbol{p}_i$ is the angle of vertex $\boldsymbol{p}_i$ of the triangle, then $\tau'(P) = \boldsymbol{p}_1$.
2. Otherwise, $\tau'(P) = g(P)$.

Let $\Phi = \{\tau\}$ and $\Phi' = \{\tau'\}$. Then $\alpha(\Phi) = \alpha(\Phi') = 1$. Sets Φ and Φ' are compatible with respect to the fault tolerant $(3,2)$-convergence problem to a line segment, but $\Phi \cup \Phi'$ is not.

7 FC(f)-PO for $f \geq 2$

This section investigates the fault tolerant (n, f)-convergence problem to f points (FC(f)-PO) for $f \geq 2$. At a glance, FC(f)-PO looks to have properties similar to FC(f), and readers might consider that the former would be easier than the latter, since in the former, the non-faulty robots are not requested to converge to a point. On the contrary, we shall see that FC(f)-PO is a formidable problem even if $f = 2$.

7.1 Compatibility

We show a difference between FC(f) and FC(f)-PO for $f \geq 2$.

Theorem 9. *Let $f \geq 2$. Any target function ϕ is not an algorithm for FC(f)-PO, if $0 \leq \alpha(\phi) < 1$, or equivalently, Φ is not compatible with respect to FC(f)-PO, if $0 \leq \alpha(\Phi) < 1$.*

Recall that $\Phi = \{\xi_{(\alpha,n)}\}$ (or $\Phi' = \{\xi'_{(\alpha,n)}\}$) is compatible with respect to FC(f) for all $2 \leq f \leq n-1$ and $0 \leq \alpha < 1$ by Theorem 7. Since $\alpha(\Phi) = \alpha$, by Theorem 9, we have:

Corollary 3. *Neither Φ nor Φ' is compatible with respect to FC(f)-PO, for all $f \geq 2$ and $0 \leq \alpha < 1$.*

7.2 Algorithm for FC(2)-PO

In Section 7.1, we showed that, for any $f \geq 2$, there is no FC(f)-PO algorithm whose scale is less than 1. It is a clear difference between FC(f)-PO and FC(f), which is solved, e.g., by CoG$_\alpha$ for any $0 \leq \alpha < 1$. This section proposes an algorithm $\psi_{(n,2)}$ with $\alpha(\psi_{(n,2)}) = 1$ for FC(2)-PO. Unfortunately, proposing an algorithm for FC(f)-PO for $f \geq 3$ is left as a future work.

Algorithm $\boldsymbol{\psi_{(n,2)}}$. We propose an algorithm $\psi_{(n,2)}$ to solve FC(2)-PO. Since the case $n = 3$ is easy, we assume $n \geq 4$. Algorithm $\psi_{(n,2)}$ calls another algorithm LN$_{(n,2)}$, which solves FC(2)-PO when an initial configuration P_0 is linear, i.e., when $CH(P_0)$ is a line segment.

To compare positions $\boldsymbol{p}$ and $\boldsymbol{q}$, we frequently use a lexicographic order $>$. It has however a drawback for our purpose: Suppose that $\boldsymbol{p}$ (resp. $\boldsymbol{q}$) in Z_0 is $\boldsymbol{p}^{(i)}$ (resp. $\boldsymbol{q}^{(i)}$) in Z_i. Then $\boldsymbol{p}^{(i)} < \boldsymbol{q}^{(i)}$ and $\boldsymbol{p}^{(j)} > \boldsymbol{q}^{(j)}$ can happen for some $i \neq j$. In $\psi_{(n,2)}$, we introduce an order $\succ$ that all robots can consistently compute.

Let $P = \{\boldsymbol{p}_1, \boldsymbol{p}_2, \ldots, \boldsymbol{p}_n\}$ be a multiset of n points, and $\overline{P} = \{\boldsymbol{q}_1, \boldsymbol{q}_2, \ldots, \boldsymbol{q}_m\}$ be the set of distinct points in P. The multiplicity of a point $\boldsymbol{q} \in \overline{P}$ is denoted by $\mu_P(\boldsymbol{q})$. In the definition of $\succ_P$, it is convenient to treat $\mu_P(\boldsymbol{q})$ points $\boldsymbol{q}$ in P as a point $\boldsymbol{q}$ with a label $\mu_P(\boldsymbol{q})$. We thus identify P with a pair $(\overline{P}, \mu_P)$, where μ_P is a labeling function to associate label $\mu_P(\boldsymbol{q})$ with each point $\boldsymbol{q} \in \overline{P}$. Let $\boldsymbol{o}_P$ be the center of the smallest enclosing circle C_P of P.

Let G_P be the rotation group $G_{\overline{P}}$ of $\overline{P}$ about $\boldsymbol{o}_P$ preserving μ_P. The order $|G_P|$ of G_P is denoted by k_P. Note that k_P does not depend on $\mu_P(\boldsymbol{o}_P)$. It is similar to the symmetricity $\sigma(P)$ of P defined in [18], but $k_P \neq \sigma(P)$ in general. Let $\Gamma_P(\boldsymbol{q}) \subseteq \overline{P}$ be the orbit of G_P through $\boldsymbol{q} \in \overline{P}$. Then $|\Gamma_P(\boldsymbol{q})| = k_P$ for all $\boldsymbol{q} \in \overline{P} \backslash \{\boldsymbol{o}_P\}$, and $\mu_P(\boldsymbol{q}') = \mu_P(\boldsymbol{q})$ for all $\boldsymbol{q}' \in \Gamma_P(\boldsymbol{q})$. If $\boldsymbol{o}_P \in \overline{P}$, $\Gamma_P(\boldsymbol{o}_P) = \{\boldsymbol{o}_P\}$. Let $\Gamma_P = \{\Gamma_P(\boldsymbol{q}) : \boldsymbol{q} \in \overline{P}\}$ be the set of all orbits. Then Γ_P is a partition of $\overline{P}$.

To define $\succ_P$, we need the concept of *view*. Define an x-y coordinate system $\Xi_{\boldsymbol{q}}$ for any point $\boldsymbol{q} \in \overline{P} \setminus \{\boldsymbol{o}_P\}$. The origin of $\Xi_{\boldsymbol{q}}$ is $\boldsymbol{q}$, the unit distance is the radius of C_P, and the x-axis is taken so that it goes through $\boldsymbol{o}_P$ in its positive side. Finally, it is right-handed. Let $\gamma_{\boldsymbol{q}}$ be the coordinate transformation from $\Xi_{\boldsymbol{q}}$ to Z_0. Then the view $V_P(\boldsymbol{q})$ of $\boldsymbol{q}$ is defined to be $\gamma_{\boldsymbol{q}}^{-1}(P)$. That is, $\gamma_{\boldsymbol{q}}(V_P(\boldsymbol{q})) = P$, i.e., P in Z_0 is $V_P(\boldsymbol{q})$ in $\Xi_{\boldsymbol{q}}$. By definition, $V_P(\boldsymbol{q}) = V_P(\boldsymbol{q}')$ if and only if $\boldsymbol{q}' \in \Gamma_P(\boldsymbol{q})$. Let $View_P = \{V_P(\boldsymbol{q}) : \boldsymbol{q} \in \overline{P} \setminus \{\boldsymbol{o}_P\}\}$.

To compare two views in $View_P$, we arbitrarily choose and fix a total order $\sqsupset$ on the set of multisets of n points. We define a total order $\succ_P$ on Γ_P as follows: For any two distinct orbits $\Gamma_P(\boldsymbol{q})$ and $\Gamma_P(\boldsymbol{q}')$ in Γ_P, $\Gamma_P(\boldsymbol{q}) \succ_P \Gamma_P(\boldsymbol{q}')$, if one of the following conditions hold: (1) $\mu_P(\boldsymbol{q}) > \mu_P(\boldsymbol{q}')$, (2) $\mu_P(\boldsymbol{q}) = \mu_P(\boldsymbol{q}')$ and $dist(\boldsymbol{q}, \boldsymbol{o}_P) < dist(\boldsymbol{q}', \boldsymbol{o}_P)$, or (3) $\mu_P(\boldsymbol{q}) = \mu_P(\boldsymbol{q}'), dist(\boldsymbol{q}, \boldsymbol{o}_P) = dist(\boldsymbol{q}', \boldsymbol{o}_P)$, and $V_P(\boldsymbol{q}) \sqsupset V_P(\boldsymbol{q}')$. When $\boldsymbol{o}_P \in \overline{P}$, we assume that $\Gamma_P(\boldsymbol{q}) \succ_P \Gamma_P(\boldsymbol{o}_P)$ for all $\boldsymbol{q} \neq \boldsymbol{o}_P$. Now $\succ_P$ is a total order on Γ_P. If $k_P = 1$, since $\Gamma_P(\boldsymbol{q}) = \{\boldsymbol{q}\}$ for all $\boldsymbol{q} \in \overline{P}$, we regard $\succ_P$ as a total order on $\overline{P}$ (by identifying $\Gamma_P(\boldsymbol{q})$ with $\boldsymbol{q}$).

We partition the set of all multisets $P = \{\boldsymbol{p}_1, \boldsymbol{p}_2, \ldots, \boldsymbol{p}_n\}$ for all $n \geq 4$ into six types G, L, T, I, S, and Z. Let $m_P = |\overline{P}|$.

G(oal): $m_P \leq 2$.
L(ine): $CH(P)$ is a line segment.
T(riangle): $m_P = 3$ and $CH(P)$ is a triangle.
I(nside): $m_P = 4$, $CH(P)$ is a triangle, and $\boldsymbol{o}_P \in P$.
S(ide): $m_P = 4$, $CH(P)$ is a triangle, and $\boldsymbol{M}_P \in P$, where $\boldsymbol{M}_P$ is the middle point of a longest side of $CH(P)$.
Z: P does not belong to the above five types.

We define a target function $\psi_{(n,2)}$.

[Target function $\psi_{(n,2)}$]

1. When P is type Z:
   (a) If $k_P \geq 2$, $\psi_{(n,2)}(P) = \boldsymbol{o}_P$.
   (b) If $k_P = 1$, $\psi_{(n,2)}(P) = \boldsymbol{a}_P$, where $\boldsymbol{a}_P \in \overline{P}$ is the largest point with respect to $\succ_P$, which is well-defined since $k_P = 1$.
2. If P is type L, invoke $\mathrm{LN}(n, 2)$.
3. When P is type T, let $\overline{P} = \{\boldsymbol{a}, \boldsymbol{b}, \boldsymbol{c}\}$.
   (a) If triangle $\boldsymbol{abc}$ is equilateral, $\psi_{(n,2)}(P) = \boldsymbol{o}_P$.
   (b) If triangle $\boldsymbol{abc}$ is not equilateral, $\psi_{(n,2)}(P) = \boldsymbol{M}_P$, where $\boldsymbol{M}_P$ is the middle point of the longest side. If there are two longest sides, $\boldsymbol{M}_P$ is the middle point of the side next to the shortest side counter-clockwise.
4. If P is type I, $\psi_{(n,2)}(P) = \boldsymbol{o}_P$.

5. If P is type S, $\psi_{(n,2)}(P) = \boldsymbol{M}_P$ (which is defined in the definition of type S).

Algorithm $\mathbf{LN}_{(n,2)}$. We present target function $\mathrm{LN}_{(n,2)}$. Let $P = \{\boldsymbol{p}_1, \boldsymbol{p}_2, \ldots, \boldsymbol{p}_n\} \in \mathcal{P}$ be a configuration of type L, which may be a configuration that a robot identifies in Look phase. We identify a point $\boldsymbol{p}_i$ in R^2 with a point in R: Since $(0,0) \in P$, we rotate P about $(0,0)$ counter-clockwise so that the resultant P becomes the multiset of points in the x-axis. Then we denote $(p,0)$ by p. In what follows in this section, a configuration P is thus regarded as a multiset of n real numbers, including at least one 0. We assume $p_1 \leq p_2 \leq \cdots \leq p_n$. By $\overline{P} = \{b_1, b_2, \ldots, b_{m_P}\}$, we denote the set of distinct real numbers in P, where m_P is the size $|\overline{P}|$ of $\overline{P}$, and $b_1 < b_2 < \cdots < b_{m_P}$. The length of $CH(P)$ is denoted by $L_P = b_{m_P} - b_1 = p_n - p_1$. Let $\lambda_P = \max_{p \in P} \min\{p - p_1, p_{m_P} - p\} \leq L_P/2$. Define j^* by $b_{j^*} = 0$. (Thus the current position of a robot r_i who identifies P in Look phase is b_{j^*} in Z_i.) Since P is type L, $k_P \leq 2$. We denote the middle point of x and y by M_{xy}, i.e., $M_{xy} = (x+y)/2$.

Like $\psi_{(n,2)}$, we consider 10 types, which we define as follows:

G: $m_P \leq 2$.
B_3: $m_P = 3$ and $k_P = 2$.
B_4: $m_P = 4$ and $k_P = 2$.
B_5: $m_P = 5$ and $k_P = 2$.
B_6: $m_P = 6$ and $k_P = 2$.
B: $m_P \geq 7$ and $k_P = 2$.
U_3: $m_P = 3$ and $k_P = 1$.
W: $m_P = 4$, $k_P = 1$, and $\overline{P} = \{b_1, b_2, b_3, b_4\} (b_1 < b_2 < b_3 < b_4)$ satisfies either (a) $2(b_2 - b_1) = b_3 - b_2$ and $b_3 \leq M_{b_1 b_4}$, or (b) $2(b_4 - b_3) = b_3 - b_2$ and $b_2 \geq M_{b_1 b_4}$.
U_4: $m_P = 4$, $k_P = 1$, and P is not type W.
U: $m_P \geq 5$ and $k_P = 1$.

We now give the target function $\mathrm{LN}_{(n,2)}$.

[Target function $\mathbf{LN}_{(n,2)}$]

1. If P is type G, $\mathrm{LN}_{(n,2)}(P) = 0$.
2. When P is type B: If $j^* \leq \lceil m_P/2 \rceil$, $\mathrm{LN}_{(n,2)}(P) = b_1$. Otherwise if $j^* > \lceil m_P/2 \rceil$, $\mathrm{LN}_{(n,2)}(P) = b_{m_P}$.
3. When P is type B_3: $m_P = 3$. If $j^* \leq 2$, $\mathrm{LN}_{(n,2)}(P) = M_{b_1 b_2}$. Otherwise if $j^* = 3$, $\mathrm{LN}_{(n,2)}(P) = M_{b_2 b_3}$.
4. When P is type B_4: $m_P = 4$. If $j^* \leq 2$, $\mathrm{LN}_{(n,2)}(P) = M_{b_1 b_2}$. Otherwise if $j^* \geq 3$, $\mathrm{LN}_{(n,2)}(P) = M_{b_3 b_4}$.
5. When P is type B_5: $m_P = 5$. If $j^* \leq 3$, $\mathrm{LN}_{(n,2)}(P) = b_2$. Otherwise if $j^* \geq 4$, $\mathrm{LN}_{(n,2)}(P) = b_4$.
6. When P is type B_6: $m_P = 6$. If $j^* \leq 3$, $\mathrm{LN}_{(n,2)}(P) = b_2$. Otherwise if $j^* \geq 4$, $\mathrm{LN}_{(n,2)}(P) = b_5$.
7. When P is type U: Since $k_P = 1$, either $b_1 \succ_P b_{m_P}$ or $b_{m_P} \succ_P b_1$ holds. If $b_1 \succ_P b_{m_P}$ then $\mathrm{LN}_{(n,2)}(P) = b_1$. Otherwise if $b_{m_P} \succ_P b_1$, $\mathrm{LN}_{(n,2)}(P) = b_{m_P}$.

8. When P is type U_3: Since $k_P = 1$ and $m_P = 3$, if $b_2 = M_{b_1 b_3}$, then $\mu_P(b_1) \neq \mu_P(b_3)$. If $b_2 < M_{b_1 b_3}$ or $(b_2 = M_{b_1 b_3}) \wedge (\mu_P(b_1) > \mu_P(b_3))$, then $\mathrm{LN}_{(n,2)}(P) = (2b_1 + b_2)/3$. Otherwise, if $b_2 > M_{b_1 b_3}$ or $(b_2 = M_{b_1 b_3}) \wedge (\mu_P(b_1) < \mu_P(b_3))$, then $\mathrm{LN}_{(n,2)}(P) = (b_2 + 2b_3)/3$.
9. When P is type W: $k_P = 1$, $m_P = 4$, and P satisfies either condition (a) or (b) (of the definition of type W).
   (a) If $2(b_2 - b_1) = b_3 - b_2$ and $b_3 \leq M_{b_1 b_4}$, then $\mathrm{LN}_{(n,2)}(P) = b_2$.
   (b) If $2(b_4 - b_3) = b_3 - b_2$ and $b_2 \geq M_{b_1 b_4}$, then $\mathrm{LN}_{(n,2)}(P) = b_3$.
10. When P is type U_4: $k_P = 1$, $m_P = 4$, and P is not type W. Suppose that $\mu_P(b_1) \geq \mu_P(b_4)$ holds. (The case P satisfies $\mu_P(b_1) < \mu_P(b_4)$ is symmetric, and we omit it.)
   (a) If $\mu_P(b_1) \geq \mu_P(b_3)$, then $\mathrm{LN}_{(n,2)}(P) = b_1$.
   (b) If $(\mu_P(b_1) < \mu_P(b_3)) \wedge (\mu_P(b_3) \geq 3)$, $\mathrm{LN}_{(n,2)}(P) = b_1$, if $b_3 = 0$, and $\mathrm{LN}_{(n,2)}(P) = 0$, otherwise if $b_3 \neq 0$.
   (c) Otherwise if $(\mu_P(b_1) < \mu_P(b_3)) \wedge (\mu_P(b_3) < 3)$, $\mu_P(b_1) = \mu_P(b_4) = 1$ and $\mu_P(b_3) = 2$. $\mathrm{LN}_{(n,2)}(P) = b_1$, if $(b_2 = 0) \vee (b_3 = 0)$, and $\mathrm{LN}_{(n,2)}(P) = 0$, otherwise if $(b_1 = 0) \vee (b_4 = 0)$.

We have the following theorem:

Theorem 10. *Target function $\psi_{(n,2)}$, which satisfies $\alpha(\psi_{(n,2)}) = 1$, is an algorithm for FC(2)-PO.*

8 Gathering Problem

We finally investigate the gathering problem, provided that there are no faulty robots, to emphasize that the gathering and the convergence problems have completely different properties from the viewpoint of compatibility. Since the gathering problem is not solvable if $n = 2$ [18], we assume $n \geq 3$ in this section. Moreover, we assume that the robots initially occupy distinct points. There are many gathering algorithms. The following algorithm GAT [18] is one of them.

[Target function GAT]
1. If there is a unique $\boldsymbol{p} \in P$ such that $\mu_P(\boldsymbol{p}) > 1$, $\mathrm{GAT}(P) = \boldsymbol{p}$.
2. Otherwise, if $\mu_P(\boldsymbol{p}) = 1$ for all $\boldsymbol{p} \in P$:
   (a) If $k_P = 1$, $\mathrm{GAT}(P) = \boldsymbol{p}$, where $\boldsymbol{p}$ is the largest point in P with respect to $\succ_P$.
   (b) If $k_P > 1$, $\mathrm{GAT}(P) = \boldsymbol{o}_P$.

Observe that $\alpha(\mathrm{GAT}) = 1$. We can modify Step 2(a) of GAT to obtain another algorithm GAT′. For example, $\mathrm{GAT}'(P)$ can be the smallest point $\boldsymbol{p}'$ in P with respect to $\succ_P$, instead of $\boldsymbol{p}$. Then indeed GAT′ is also a gathering algorithm with $\alpha(\mathrm{GAT}') = 1$, but obviously $\Phi = \{\mathrm{GAT}, \mathrm{GAT}'\}$ is not compatible with respect to the gathering problem. Let us summarize.

Theorem 11. [18] *Let $\Phi = \{\mathrm{GAT}\}$ and $\Phi' = \{\mathrm{GAT}'\}$. Then Φ and Φ' are compatible with respect to the gathering problem, but $\Phi \cup \Phi'$ is not. Here $\alpha(\Phi) = \alpha(\Phi') = \alpha(\Phi \cup \Phi') = 1$.*

Theorem 12. *Any target function ϕ is not a gathering algorithm if $\alpha(\phi) < 1$, or equivalently, any set Φ of target functions such that $\alpha(\Phi) < 1$ is not compatible with respect to the gathering problem.*

9 Conclusions

We introduced the concept of compatibility and investigated the compatibilities of several convergence problems. A compatible set Φ of target functions with respect to a problem Π is an extension of an algorithm ϕ for Π, in the sense that every target function $\phi \in \Phi$ is an algorithm for Φ.

The problems we investigated are the convergence problem, the fault tolerant (n, f)-convergence problem (FC(f)), the fault tolerant (n, f)-convergence problem to a convex f-gon (FC(f)-CP), and the fault tolerant (n, f)-convergence problem to f points (FC(f)-PO), for crash faults. The gathering problem was also investigated. The results are summarized in Table 1. Main observations we would like to emphasize are:

1. The convergence, FC(1), FC(1)-PO, and FC(f)-CP share the same property: Every set Φ of target functions is compatible, if $0 \leq \alpha(\Phi) < 1$.
2. The gathering problem and FC(f)-PO for $f \geq 2$ share the same property: Any set Φ of target functions is **not** compatible, if $0 \leq \alpha(\Phi) < 1$.
3. FC(f) ($f \geq 2$) is in between FC(f)-CP and FC(f)-PO.

Before closing the paper, we list some open problems:

1. Extend Table 1 to contain the results for $\alpha(\Phi) > 1$.
2. Suppose that ϕ and ϕ' are algorithms for the convergence problem. Find a necessary and/or a sufficient condition for $\Phi = \{\phi, \phi'\}$ to be compatible with respect to the convergence problem.
3. Investigate the compatibility of FC(f)-PO for $f \geq 2$ under the $\mathcal{FSYNC}$ model.
4. Investigate the compatibility of convergence problems under the $\mathcal{ASYNC}$ model.
5. Investigate the compatibility of convergence problems in the presence of Byzantine failures.
6. Investigate the compatibility of fault tolerant gathering problems.
7. Find interesting problems with a large compatible set.

Acknowledgments. This work is supported in part by JSPS KAKENHI Grant Numbers JP17K00024 and JP22K11915.

References

1. N. Agmon and D. Peleg, *Fault-tolerant gathering algorithms for autonomous mobile robots*, In Proc. 15th Annual ACM-SIAM Symposium on Discrete Algorithms, pp. 1063–1071, 2004.

2. H. Ando, Y. Oasa, I. Suzuki, and M. Yamashita, *A distributed memoryless point convergence algorithm for mobile robots with limited visibility*, IEEE Trans. Robotics and Automation, 15, 818–828, 1999.
3. Y. Asahiro, I. Suzuki, and M. Yamashita, *Monotonic self-stabilization and its application to robust and adaptive pattern formation*, Theoretical Computer Science, 934, 21–46, 2022.
4. Y. Asahiro and M. Yamashita. *Compatibility of convergence algorithms for autonomous mobile robots* arXiv: 2301.10949, 2023
5. Z. Bouzid, S. Das, and S. Tixeuil, *Gathering of mobile robots tolerating multiple crash faults*, In Proc. IEEE 33rd Int'l Conference on Distributed Computing Systems, pp. 337–346, 2013.
6. K. Buchin, P. Flocchini, I. Kostitsyana, T. Peters, N. Santoro, and K. Wada, *On the computational power of energy-constrained mobile robots: Algorithms and cross-model analysis*, In Proc. International Colloquium on Structural Information and Communication Complexity, pp. 42–61, 2022.
7. M. Cieliebak, P. Flocchini, G. Prencipe, and N. Santoro, *Distributed computing by mobile robots: gathering*, SIAM J. Computing, 41, 829–879, 2012.
8. R. Cohen and D. Peleg, *Convergence properties of the gravitational algorithm in asynchronous robot systems*, SIAM J. Computing, 34, 1516–1528, 2005.
9. R. Cohen and D. Peleg, *Convergence of autonomous mobile robots with inaccurate sensors and movements*, SIAM J. Computing, 38, 276–302, 2008
10. A. Cord-Landwehr, B. Degener, M. Fischer, M Hüllman, B. Kempkes, A. Klaas, P. Kling, S. Kurras, M. Märtens, F. Meyer auf der Heide, C. Raupach, K. Swierkot, D. Warner, C. Weddemann, and D. Wonisch, *A new approach for analyzing convergence algorithms for mobile robots*, In Proc. ICALP 2011, Part II, LNCS, vol. 6756, pp. 650–661, 2011.
11. S. Das, P. Flocchini, N. Santoro, and M Yamashita, *Forming sequences of geometric patterns with oblivious mobile robots*, Distributed Computing, 28, 131–145, 2015.
12. X. Défago, M. G. Potop-Butucaru, and S. Tixeuil, *Fault-tolerant mobile robots*, In *Distributed Computing by Mobile Entites*, LNCS, vol. 11340, pp. 234–251, 2019.
13. P. Flocchini, *Gathering*, In *Distributed Computing by Mobile Entites*, LNCS, vol. 11340, pp. 63–82, 2019.
14. P. Flocchini, G. Prencipe, and N. Santoro, *Distributed Computing by Oblivious Mobile Robots,* Synthesis Lectures on Distributed Computing Theory 10, Morgan & Claypool Publishers (2012).
15. T. Izumi, S. Soussi, Y. Katayama, N. Inuzuka, X. Defago, K. Wada, and M. Yamashita, *The gathering problem for two oblivious robots with unreliable compasses*, SIAM J. Computing, 41, 26–46, 2012.
16. B. Katreniak, *Convergence with limited visibility by asynchronous mobile robots*, In Proc. 15th Int'l Symposium on Stabilization, Safety, and Security of Distributed Systems, pp. 125–137, 2011.
17. G. Prencipe, *Pattern formation*, In *Distributed Computing by Mobile Entites*, LNCS, vol.11340, pp. 37–62, 2019.
18. I. Suzuki and M. Yamashita, *Distributed anonymous mobile robots – formation and agreement problems*, SIAM J. Computing, 28, 1347–1363, 1999.
19. M. Yamashita and I. Suzuki, *Characterizing geometric patterns formable by oblivious anonymous mobile robots*, Theoretical Computer Science 411, 2433–2453, 2010.
20. Y. Yamauchi, T. Uehara, S. Kijima, and M. Yamashita, *Plane formation by synchronous mobile robots in the three-dimensional Euclidean space*, J. ACM, 64, 1–43. 2017