# Distributed Pattern Formation With Faulty Robots

Debasish Pattanayak[†]
*Department of Mathematics*
*IIT Guwahati*
Guwahati, India
p.debasish@iitg.ac.in

Klaus-Tycho Foerster
*Faculty of Computer Science*
*University of Vienna*
Vienna, Austria
klaus-tycho.foerster@univie.ac.at

Partha Sarathi Mandal
*Department of Mathematics*
*IIT Guwahati*
Guwahati, India
psm@iitg.ac.in

Stefan Schmid
*Faculty of Computer Science*
*University of Vienna*
Vienna, Austria
stefan_schmid@univie.ac.at

arXiv:2003.01914v1 [cs.DC] 4 Mar 2020

***Abstract*—Pattern formation is one of the most fundamental problems in distributed computing, which has recently received much attention. In this paper, we initiate the study of distributed pattern formation in situations when some robots can be *faulty*. In particular, we consider the usual and challenging *look-compute-move* model with oblivious, anonymous robots. We first present lower bounds and show that any deterministic algorithm takes at least two parallel rounds to form simple patterns in the presence of faulty robots. We then present distributed algorithms for our problem which match this bound, *for conic sections*: in two parallel rounds, robots form lines, circles, parabola, and ellipses while tolerating $f = 2$, $f = 3$, $f = 4$, and $f = 5$ faults, respectively. We show that the resulting pattern includes the $f$ faulty robots in the pattern of $n$ robots, where $n \geq 2f + 1$, and that $f < n < 2f + 1$ robots cannot form such patterns. We conclude by discussing several relaxations and extensions.**

***Index Terms*—Distributed Algorithms, Pattern Formation, Fault Tolerance**

## I. Introduction

Self-organizing systems have fascinated researchers for many decades already. These systems are capable of forming an overall order from an initially disordered configuration, using *local* interactions between its parts only. Self-organization arises in many forms, including physical, chemical, and biological systems, and can be based on various processes, from crystallization, over chemical oscillation, to neural circuits [7]. Due to their decentralized and self-healing properties, self-organizing systems are often very robust.

We, in this paper, consider self-organizing systems in the context of *robotics*. In particular, we are interested in the fundamental question how most simple robots can self-organize into basic patterns. This *pattern formation* problem has already received much attention in the literature [6], [16], [21], [22]. A particularly well-studied and challenging model is known as the *look-compute-move* model, in which each robot, in each round, first observes its environment and then decides on its next move. In the most basic setting, the robots do not have any memory and hence cannot remember information from previous rounds, and they can also not distinguish between the other robots they see: the robots are *oblivious* and *anonymous*. Furthermore, robots cannot communicate. Over the last years, several interesting results have been obtained on when robots can and cannot form different patterns [6], [13], [16], [18], [21], [22], [24], [25]. However, existing work on pattern formation shares the assumption that robots are non-faulty.

This paper initiates the study of distributed pattern formation algorithms for scenarios where some robots can be *faulty*: the faulty robots do not move nor act according to a specified protocol or algorithm. The setting with faulty robots is particularly challenging, as the non-faulty robots cannot directly observe which robots are faulty. But even *indirect* observations seem challenging: since all robots are oblivious, a robot cannot remember patterns from previous rounds, and hence, has no information which robots moved recently and which not. In fact, a robot *per se* does not even know whether the current pattern it observes is the initial configuration or whether some rounds of movements already occurred. What's more, the ability to self-organize into a specific pattern seems to require some coordination or even consensus, which is notoriously hard in faulty environments.

### A. Our Contributions

This paper considers the design of distributed algorithms for pattern formation of most simple oblivious and potentially *faulty* robots. Our main result is an algorithmic framework that allows robots to form patterns which include faulty robots, in a decentralized and efficient manner. In particular, we do not require robots to explicitly identify faulty robots or be able to remember previous configurations.

Depending on the number of faults, $f \in \{2, 3, 4, 5\}$, we show how to form specific target patterns such as the line, circle, parabola, and ellipse, respectively, in just *two rounds*, for at least $2f + 1$ robots. We also prove that this is optimal: no deterministic algorithm can solve this problem in just one round or with less than $2f + 1$ robots.

We further discuss several relaxations of our model and extensions of our results, e.g., considering initial symmetric configurations, having at most $f$ faulty robots, or the impossibility of forming the pattern corresponding to $f$ faults.

### B. Related Works

Pattern formation is an active area of research [13], [18], [25], however, to the best of our knowledge, we are the first to consider pattern formation in the presence of faults: a fundamental extension. In general, pattern formation allows for exploring the limits of what oblivious robots can compute.

[†]Visit to University of Vienna is supported by the Overseas Visiting Fellowship, 2018 Award No. ODF/2018/001055 by the Science and Engineering Research Board (SERB), Government of India.

A "weak robot" model was introduced by Flocchini et al. [16]. The objective was to determine the minimum capabilities a set of robots need to achieve certain tasks. In general, the tasks include *Gathering* [8]–[10], [15], *Convergence* [11], [12], *Pattern Formation* [6], [18], [21], etc. Gathering is a special case of pattern formation, where the target pattern is a point. Gathering has been achieved for robots with multiplicity detection [10], [17]. Most gathering algorithms use the capability of multiplicity detection to form a unique multiplicity point starting from a scattered configuration. In the absence of this capability, it has been proved that gathering is impossible in the semi-synchronous model [20].

The objective of gathering algorithms is only to gather the non-faulty robots, not to form general patterns. And, for the specific case of gathering, some interesting first fault-tolerance studies exist. Agmon and Peleg [1] solve the gathering problem for a single crash-fault. Gathering has been solved with multiple faults [4], [5] with strong multiplicity detection. Next, gathering has also been addressed for robots with weak multiplicity detection tolerating multiple crash faults [3], [19]. For byzantine faults, Auger et al. [2] show impossibility results, and Defago et al. [14] present a self-stabilizing algorithm for gathering. The gathering algorithms only gather non-faulty robots. Since oblivious robots cannot differentiate a faulty robot from a non-faulty one, all the algorithms can be considered to be non-terminating algorithms. In contrast in our paper, we include the faulty robots in the pattern, and as a result, we achieve termination.

Flocchini et al. [16] characterized the role of common knowledge, like agreement on the coordinate system as a requisite for the pattern formation problem, and Yamashita and Suzuki [23] characterized the patterns formable by oblivious robots. Fujinaga et al. [18] presented an algorithm using bipartite matching for asynchronous robots. Yamauchi and Yamashita [25] proposed a pattern formation algorithm for robots with limited visibility. Das et al. [13] characterize the sequence of patterns that are formable, starting from an arbitrary configuration. Formation of a plane starting from a three-dimensional configuration has also been solved for synchronous robots [22], [24]. The authors characterize the configurations for which plane formation is possible. The existing pattern formation algorithms consider the sequential movement of robots. Since we consider faulty robots in our paper, all the existing algorithms are not adaptable to our cause. A fault-tolerant algorithm has to consider parallel robot movement and should satisfy the wait-free property.

### C. Organization

The remainder of our paper is structured as follows. We first provide a formal model in §II, followed by a study of the special case of $f = 1$ faulty robot in §III to provide some intuition. We then give tight runtime and cardinality lower bounds in §IV, and match them for the remaining conic patterns in §V. After discussing further model variations in §VII, we conclude in §VIII.

## II. Preliminaries

We follow standard model assumptions, inspired by existing work such as, e.g., [1], [3], [19].

### A. Model

The robots are homogeneous: they execute the same deterministic algorithm, are indistinguishable and anonymous (no identifiers), oblivious (no persistent memories), and silent (no communication). The robots do not share a common coordinate system and observe others in their own *local coordinate system*. The robots have unlimited visibility, and the determined locations of other robots are precise. Each robot follows the *look-compute-move* cycle. A robot obtains a snapshot of relative positions of other robots with respects to its position in the *look* state. Based on the snapshot of other robot positions, it decides a destination in the *compute* state. In the *move* state, it moves to the destination and reaches the destination in the same round. This is known as *rigid* robot movement. The scheduler, which activates the robots, follows a fully-synchronous (*FSYNC*) model, i.e., all the robots look at the same time and finish movement in the same round. We consider that the robots are susceptible to crash-faults, i.e., they stop moving after the crash and never recover. Moreover, the number of $f$ faulty robots is known beforehand, and as such, the robots know which type of pattern to form. In particular, we assume that the following four initial conditions hold:

1) All $f$ faulty robots have already crashed initially.
2) All initial configurations are asymmetric.[1]
3) All robots occupy distinct positions initially.[2]
4) The faulty robots form a convex polygon.

The last assumption needs the faulty robots to be at corners of a convex polygon; the non-faulty robots can lie at any position. The rationale behind the assumption is due to the fact that four robots forming a triangle with a robot inside the triangle do not correspond to any conic section where all the robots lie on the conic section. Similarly, for five robots. For three or more robots, a collinear configuration is addressed in §VII. For two robots, the assumption trivially holds.

### B. Notations

A configuration $\mathcal{C} = \{p_1, p_2, \cdots, p_n\}$ is a set of $n$ points on the plane, where $p_i = (x_i, y_i)$ is a tuple representing the $x$- and $y$-coordinates of the robots. Since each robot is initially located at distinct points, it then holds that $p_i \neq p_j$ for any pair of $i$ and $j$ such that $i \neq j$. $f$ is the number of faulty robots. We will always uphold this condition in our algorithms, except for the case of $f = 1$, where the target pattern is a point.

## III. Problem Statement and Intuition

### A. Objective

Given a set of robots on the plane as defined in the model (§II-A), we want the robots to achieve two objectives:

[1]This assumption allows us to have a unique ordering of the robots [9].

[2]As any set of non-faulty robots that share a position will always perform the same actions from then on and be indistinguishable from each other.

1) According to the number of faults, the robots should form a corresponding conic pattern.[3]
2) The robots have to achieve a quasi-uniform pattern with the non-faulty robots, by virtue of which the faulty robots can be identified from their positions on the plane.

### B. Point Formation ($f = 1$)

In order to provide some intuition, we start with the simpler case of $f = 1$. For a single faulty robot, we move all the robots to the center of their smallest enclosing circle. If there is a faulty robot in the center, then point formation is achieved. If the faulty robot is somewhere else, we arrive at a configuration with two robot positions. For a configuration with two robot positions, all robots move to the other robots' position. From Assumption 3, we have all the robots at distinct initial positions. Hence, the robot which cannot move is at a different position from the gathered robots. Moving all the robots to the other robots' position achieves our objective of point formation since the faulty robot cannot move.

## IV. Lower bound

From our initial example in §III-B, we saw that there could be situations where one parallel round suffices, namely, for the case of exactly $2f = 2$ robots. However, we can show that for $f \geq 2$, at least two rounds are required. For conic patterns (with $f \in \{2, 3, 4, 5\}$), this bound is tight: we will later provide algorithms that finish in two rounds.

**Theorem 1.** *For every $f \geq 2$ and every $n \geq f+3$ holds: Any deterministic algorithm needs more than one round to make a pattern passing through all $f$ faulty robots.*

*Proof.* Let $\varphi$ be a deterministic algorithm that results in a pattern with faults. Suppose $\varphi$ solves the pattern in one step. Two patterns can have at most $f$ common points[4]. Let $\mathcal{C} = \{p_1, p_2, \cdots, p_{f+3}\}$ be an initial configuration with $f+3$ robots such that no $f+1$ robots are in the same pattern.

Without loss of generality, consider two sets of $f$ faulty robots at positions $\{p_1, p_2, \cdots, p_f\}$ and $\{p_2, p_3, \cdots, p_{f+1}\}$ and the corresponding pattern be $\wp$ and $\wp'$, respectively. The $f$ faulty robots do not move. Let $\wp = \{p'_1, p'_2, \cdots, p'_{f+3}\}$ and $\wp' = \{p''_1, p''_2, \cdots, p''_{f+3}\}$. As $\varphi$ achieves pattern formation in one round, both $\wp$ and $\wp'$ should be final. We have $p_{f+1} \neq p'_{f+1}$, $p_{f+2} \neq p'_{f+2}$ and $p_{f+3} \neq p'_{f+3}$, since all robots in $\wp$ are in the pattern. Similarly, we also have $p_1 \neq p''_1$, $p_{f+2} \neq p''_{f+2}$ and $p_{f+3} \neq p''_{f+3}$ for $\wp'$. Since the robots at $\{p_2, \cdots, p_f\}$ did not move to form $\wp$ and $\wp'$, these $f-1$ points are common between $\wp$ and $\wp'$. Since, $p_{f+1} \neq p'_{f+1}$ and $p_{f+1} = p''_{f+1}$, so $p_{f+1}$ cannot be a common point between $\wp$ and $\wp'$. Out of $p_{f+2}$ and $p_{f+3}$, at most one can be a common point in the pattern, since there are at most $f$ common points. Since, $\varphi$ is deterministic, the destination for robots at $p_{f+2}$ and $p_{f+3}$ remains the same regardless of the destination pattern being $\wp$ or $\wp'$. So, $p'_{f+2} = p''_{f+2}$ and $p'_{f+3} = p''_{f+3}$, thus $\wp$ and $\wp'$ have $f+1$ common points. This is a contradiction since the patterns are different. Hence no deterministic algorithm can solve the pattern formation problem with faults in one round. The arguments hold analogously for $n \geq f+3$ robots. □

Next, we show a lower bound on the number of robots required to solve the pattern formation problem. A configuration with exactly $f$ robots is trivially solvable since the $f$ robots are already in the pattern. Note that for $f \geq 2$, $2f+1 \geq f+3$.

**Theorem 2.** *At least $2f+1$ robots are required to form a pattern passing through $f$ faulty robots for $f \geq 2$.*

*Proof.* Consider the number of robots to be $f < n < 2f+1$. Assume that the configuration of these $n$ robots is such that no $f+1$ robots are in the same pattern. Let $\varphi$ be a deterministic algorithm, which decides the destination of the robots given a configuration. Since the robots are oblivious, it is impossible to determine which robots are faulty given a configuration. As we consider patterns from the conic section, a pattern corresponding to $f$ can be uniquely determined through $f$ robots.

Let $\varphi$ decide the target pattern corresponding to a set of $f$ robots for the given configuration $\mathcal{C}$. So the other $n-f$ robots have to move to the pattern. Since the algorithm cannot determine which robots are faulty, the adversary can always choose the $0 < n-f$ robots to be faulty. Since $n-f \leq f$, none of the robots move. This leads to a stagnated configuration, and the algorithm does not proceed further.

If the algorithm decides a pattern that passes through less than or equal to $f$ points in the configuration, then we choose faulty robots out of the points which are not on the pattern, and we arrive at a configuration where not all the robots are in the same pattern. Now there are at most $n-f \leq f$ robots on the pattern, which is the same as the previous configuration. □

## V. Detailed Algorithms

In this section, we provide the promised two round algorithms for the line (§V-B, $f = 2$), the circle (§V-C, $f = 3$), the parabola (§V-D, $f = 4$), and the ellipse (§V-E, $f = 5$). To this end, we first provide algorithmic preliminaries in §V-A.

### A. Algorithmic Preliminaries

In our algorithms, we will perform case distinctions according to the following four types of configurations:

**Definition 1** (*Terminal Configuration*)**.** *A configuration is a terminal configuration if all the robots form the target pattern corresponding to $f$ faulty robots.*

**Definition 2** (*Type I*)**.** *If exactly $n-f$ robots are in the target pattern corresponding to $f$ faulty robots, then it is a Type I configuration.*

A Type I configuration can be symmetric or asymmetric.

**Definition 3** (*Type O*)**.** *If a configuration is not Terminal or Type I, then it is a Type O configuration.*

[3] That is a point for $f = 1$, a line for $f = 2$, a circle for $f = 3$, a parabola for $f = 4$, and an ellipse for $f = 5$.

[4] Two parabolas intersect at 4 points, which can be the common points between two parabola patterns.

Note that an initial asymmetric configuration can be a Type I or Type O or Terminal configuration.

**Definition 4** (*Uniform Configuration*)**.** *A configuration is a uniform configuration if the distance between all consecutive pairs of robots in the configuration along the pattern is the same.*

**Definition 5** (*Quasi-Uniform Configuration*)**.** *If a uniform configuration with $m$ uniform positions is occupied by $n$ robots, where $n \leq m \leq 2n$, then it is a quasi-uniform configuration.*

With the assumption (§II-A) that the initial configuration is asymmetric, we can obtain an ordering of the robots using the ordering algorithm by Chaudhuri et al. [9].

**Lemma 3.** *An asymmetric configuration is orderable. [9]*

Using Lemma 3, we can thus always obtain an ordering among the robots. We use this ordering to determine the target pattern in case of a Type O configuration. In general, having an ordering allows us to have complete agreement on the coordinate system, i.e., the robots agree on the direction and orientation.

Let $\mathcal{O}$ be an ordering of the robots that maps the set of robots to a set of integers $\{1, 2, \cdots, n\}$ such that each robot corresponds to an integer. This is the rank of the robot in the ordering. In case of symmetry, two robots can have different orderings. We note that locally, the ordering is unique for a particular robot, but that from a global perspective, different robots can have different orderings.

We use the ordering to determine a target pattern only in cases where there are multiple potential target patterns. We always choose the target pattern passing through the smaller ranked robot in the ordering.

Since the algorithm takes at most two steps to reach a pattern containing all faulty robots, we denote the *initial*, *transitional* and *final* configurations as $\mathcal{C}_0$, $\mathcal{C}_1$ and $\mathcal{C}_2$, respectively.

### *B. Line Formation ($f = 2$)*

We have three different cases depending on the current configuration. An initial configuration $\mathcal{C}_0$ can be an asymmetric Type I or a Type O. A transitional configuration $\mathcal{C}_1$ can be a symmetric Type I configuration. If the pattern is already formed in $\mathcal{C}_0$ or the algorithm forms the pattern in the first step, then the objective is achieved.

*a) Type O Configuration:* For a Type O configuration, we order the robots using the algorithm from [9]. According to the ordering, we choose the robot with smallest rank which is not at the center of the circle. Then we take a line perpendicular to the diameter passing through that robot as the target line, which is at a distance of the diameter from the center on the side of the robot. In Fig. 1, the robot at $A$ is the robot with smallest rank and it is not at the center. The point $B$ is at a distance of $d$, the diameter of the smallest enclosing circle (SEC) from the center along the extended radius passing through $A$. After the line is determined, we

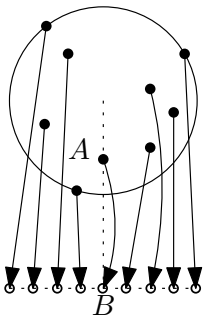


Fig. 1. Target pattern for Type O configuration

determine the destination points of the robots on the line. For an asymmetric configuration, the destination points are according to the ordering. The length of the target pattern, i.e., a line, is $\mathcal{P} = d$. The target line is positioned such that the midpoint of the line lies on the extended radius. In Fig. 1, the midpoint of the target line is $B$. We take the uniform distance $c = \mathcal{P}/n$ and determine a set of uniform points on the line. The ordering of the target points is determined according to the agreed coordinate system. Fig. 1 shows the destination points for all the robots in the current configuration. In all the figures, the current robot positions are denoted by disks, and the destinations are denoted by circles. The arrows joining the robots with their destination do not represent the paths of the robots and are only meant for illustration.

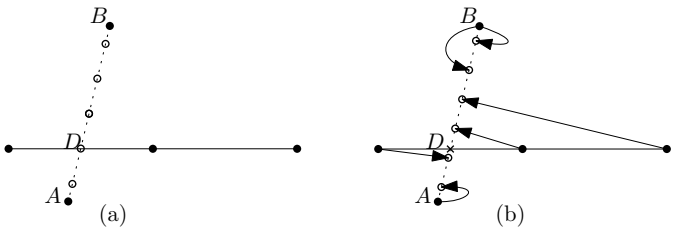


Fig. 2. Target Pattern for Asymmetric Type I Configuration

*b) Asymmetric Type I Configuration:* Given an asymmetric Type I configuration, we determine the two robots which are not present on the line with other robots. We choose the line passing through these two robots as the target line. Next, we take the length of the pattern $\mathcal{P} = d$, where $d$ is the distance between the two robots. In Fig. 2, we have $A$ and $B$ as the two robots not on the pattern in the current configuration. $|\overline{AB}| = d$. The uniform distance $c$ is $\mathcal{P}/n$. Let $A$ be the robot with smaller rank among $A$ and $B$. We choose the set of uniform points at a distance $c/2$ from $A$ towards $B$. Let $D$ be the intersection point of the existing line in the current configuration and the line through $A$ and $B$ ($D$ is marked as a cross in Fig. 2). If the uniform points overlap with $D$, then we choose another set of points at uniform distance $c' = nc/(n+1)$. In Fig. 2(a), $|\overline{AD}| = 3c/2$, so the uniform points overlap with $D$. We have to choose a set of uniform points corresponding to $c'$ as shown in Fig. 2(b). We now show the destinations for all the robots. The robots at $A$ and

$B$ move inwards since all $n$ points are contained in the line $\overline{AB}$. As there are $n+1$ uniform points, the robot $B$ has two potential destinations, one of which can be chosen arbitrarily. The robots which are on the existing configuration are moving towards uniform positions on the line according to their rank.

*c) Symmetric Type I Configuration:* This configuration can only occur in $\mathcal{C}_1$. The configuration can be symmetric with one line of reflective symmetry or a rotational symmetry with angle $\pi$. First we address the case for reflective symmetry. Given a line that is symmetric over the existing pattern, we need to ensure that no two points should have the same destination. We determine the ordering $\mathcal{O}$ for each group on one side of the line of line of symmetry for reflective symmetry. Regardless of the difference in ordering, we still guarantee that the destinations are distinct for each ordering. Since the configuration is symmetric, we know that the configuration is $\mathcal{C}_1$ and intermediate. The two robots not present on the line with other robots are faulty.

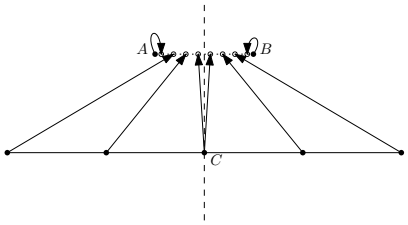


Fig. 3. A symmetric Type I configuration with reflective symmetry

Let the length of the pattern $\mathcal{P}$ be $\overline{AB} = d$. If there are $k$ robots on the line of symmetry, we choose the uniform distance $c = \mathcal{P}/(n+k)$. Similar to the previous case, choose a set of uniform points starting at a distance $c/2$ from $A$ as shown in Fig. 3. Similar to the previous case, if there is an intersection point which coincides with the uniform points then we choose uniform points again with $c' = \mathcal{P}/(n+k+1)$. Since it is a symmetric configuration, the robots on the line of symmetry have two destinations on either side of the line of symmetry. The robots choose to move to a point according to their local orientation. In Fig. 3, there is only one robot on the line of symmetry, which has two potential destinations. The robots on either side of the line of symmetry can have ordering, and choose their destinations accordingly.

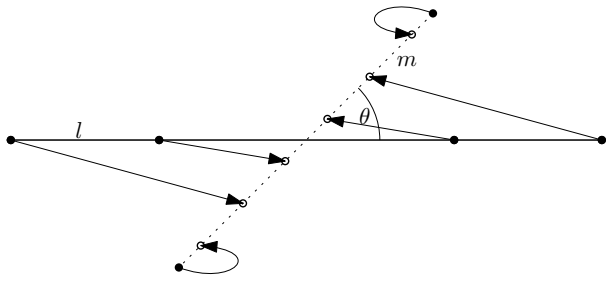


Fig. 4. Symmetric Type I configuration with rotational symmetry

In case of rotational symmetry, let $l$ be the line in the current configuration and $m$ be the line for the target configuration. Let $\theta$ be the smaller angle between $l$ and $m$. $\theta$ cannot be $\pi/2$, since this configuration results from an asymmetric Type I configuration, it would mean that the angle between the lines in the asymmetric Type I was $\pi/2$ and the midpoint of the target line was on the current line of asymmetric Type I configuration. This leads to the asymmetric Type I configuration being a configuration with reflective symmetry, a contradiction. Hence a smaller $\theta$ exists. Also, there is no robot on the intersection point, which also means that the number of robots is even. Now we choose uniform points with $c = \mathcal{P}/n$ where the pattern length $\mathcal{P}$ is the distance between the robots not on the line in the current configuration. We choose the uniform points at a distance $c/2$ from the end points of the target line. In Fig. 4, we show the robots and their destination for a configuration with rotational symmetry.

### C. Circle Formation ($f = 3$)

Similar to the previous case, we also have three different cases for the circle formation algorithm with three faults.

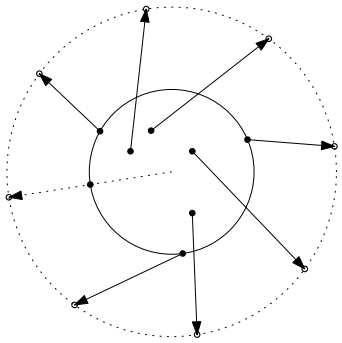

Fig. 5. Type O Configuration for $f = 3$

*a) Type O Configuration:* The robots in a Type O configuration $\mathcal{C}$ set the target pattern as a concentric circle with twice the diameter of the SEC of $\mathcal{C}$. The length of the pattern $\mathcal{P} = 2\pi d$, where $d$ is the diameter of SEC of current configuration. Take the uniform distance between points along the perimeter to be $c = \mathcal{P}/n$ (ref. Fig. 5). We have an ordering $\mathcal{O}$. We take the uniform positions on the target circle as the points uniformly starting from the point of intersection of the extended radius passing through the robot with rank one (dotted line in Fig. 5). The extended radius passing through the robot with rank two is the clockwise direction and the order of the uniform points follow the clockwise direction.

*b) Asymmetric Type I Configuration:* In case of an asymmetric Type I configuration, we can also obtain an ordering $\mathcal{O}$. The target pattern passes through the three points, which are not in the circle in the current configuration. Two circles can have at most two intersection points. Let the robots at $A$, $B$, and $C$ be the three robots that were not on the circle of the current configuration. Let $d$ be the diameter of the circle passing through $A$, $B$, and $C$. The pattern length is $\mathcal{P} = \pi d$. We choose uniform distance $c = \mathcal{P}/n$. Without loss of generality, let the order among $A$, $B$ and $C$ follow $A < B < C$. Let the intersection points of the two circles

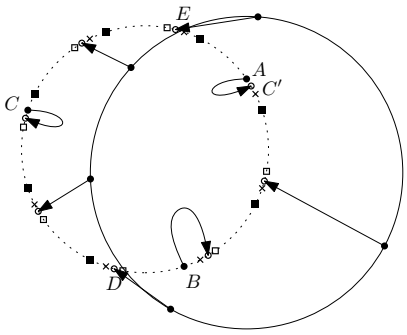


Fig. 6. Asymmetric Intermediate Configuration for $f = 3$

be $D$ and $E$ (ref. Fig. 6). We determine a set $\mathcal{U}$ of uniform points which correspond to $A$, $B$ or $C$, i.e., the set of points at a uniform distance from these points on the pattern. In Fig. 6, the uniform points corresponding to $A$, $B$, and $C$ are denoted with empty squares, filled squares, and crosses, respectively. Take the distance between $A$ and the closest point from $\mathcal{U}$, i.e., $C'$. Consider the target points as the uniform set corresponding to the midpoint of the arc $\widehat{AC'}$. Fig. 6 shows these points marked with small circles on the target circle (dotted circle). If the uniform point set includes the intersection points $D$ or $E$, we choose the second closest point from $A$. The clockwise direction is the direction of $A$ towards $B$. The ordering of the target pattern points follows the clockwise direction starting from the point near $A$. If the uniform point set fails to have a non-overlapping set of points, then we expand the uniform set $\mathcal{U}$ to include points with uniform distance $c = \mathcal{P}/(n+1)$. If there are $n+1$ destinations, then the robot with the highest rank would have two destinations similar to the algorithm described in the previous section.

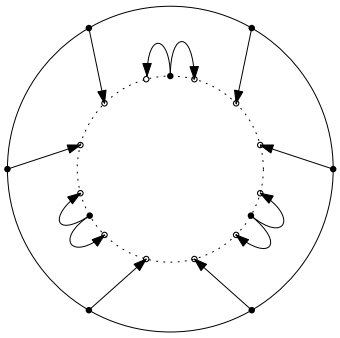

Fig. 7. Destinations for $\mathcal{C}_1$ with Rotational Symmetry

*c) Symmetric Type I Configuration:* There are two cases, reflective symmetry and rotational symmetry. For rotational symmetry, each rotation partition can be ordered uniquely with the boundary robots included in the ordering. We provide destinations for all the robots in a rotational partition and the robots on the rotational axis of symmetry. The robots on the rotational axis of symmetry would have two destinations globally, but they would choose the clockwise destination according to their local orientation. Fig. 7 shows the destinations for the robots. The arc between two rotation axes of symmetry contains four equidistant points (two for the robots on the rotation block, and two for the robots on the axes of symmetry). The points are also equidistant from the axis of symmetry. The pattern length is $\mathcal{P} = \pi d$, where $d$ is the diameter of the target circle. The uniform distance $c$ is $\mathcal{P}/(n+3)$, and the distance from the axis of symmetry is $c/2$.

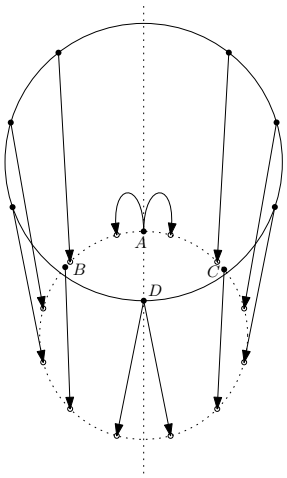


Fig. 8. A Type I configuration with reflective symmetry

For reflective symmetry, let there be $k$ robots on the line of symmetry. The points of the target pattern are chosen at an uniform distance $c = \mathcal{P}/(n+k)$ and the distance from line of symmetry is $c/2$. The $k$ robots on the line of symmetry globally have two destinations on either side of the line of symmetry. The robots choose the destination on their right side (according to local orientation). In Fig. 8, we show the destinations for the robots in a Type I reflective symmetry configuration. The robots at $A$, $B$, and $C$ are not at uniform positions, so the target circle passes through these robot positions. Since there are two robots on the line of symmetry, there are six uniform destinations on either side of the line of symmetry with two potential destinations for robots at $A$ and $D$.

### *D. Parabola Formation ($f = 4$)*

Again, we have three cases for the target pattern parabola.

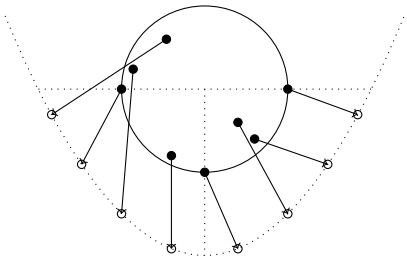

Fig. 9. Target parabola for Type O configuration

*a) Type O Configuration:* The target pattern is a parabola with the latus rectum[5] twice the diameter of the SEC of the configuration and the distance between the focus and the vertex is equal to the diameter of the SEC with the focus located at the center of the SEC. The vertex of the parabola is located on the extended radius passing through the rank one robot. Fig. 9 shows a target parabola pattern with uniform positions for a Type O configuration. Let $\mathcal{P}$ be the length of the arc of parabola between the latus rectum. We choose uniform distance $c$ to be $\mathcal{P}/n$. The distance from the latus rectum is $c/2$. Since the configuration is asymmetric, the robots can agree on a common clockwise direction and determine the destinations according to their rank in the order.

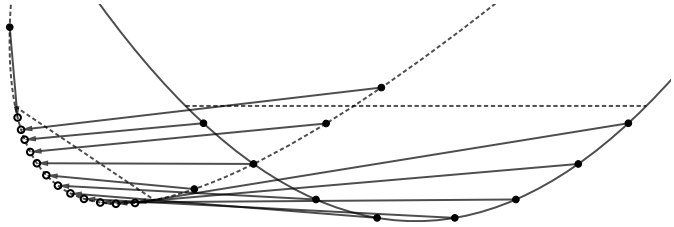

Fig. 10. An asymmetric Type I configuration for $f = 4$

*b) Asymmetric Type I Configuration:* The target parabola passes through the four robots which are not in the pattern in the current configuration. There can be two conjugate parabolas passing through the same four points. We choose the parabola with the larger latus rectum as the target pattern. We choose the uniform points between the points where the latus rectum intersects the parabola. The ordering of uniform points follows the increasing order in the clockwise direction. The uniform distance $c$ is similarly defined to be $\mathcal{P}/n$ where $\mathcal{P}$ is the length of the arc of parabola between the two intersection points of latus rectum. Choose four sets of uniform points corresponding to four points through which the target parabola passes. Let $A$ be the robot with the least rank among the four. We determine a uniform point set which does not overlap with the existing points and the intersection of two parabolas similar to the previous section with the circle. Since the parabola is an unbounded curve, we choose the uniform pattern points between the points where the latus rectum of the parabola intersects with the parabola. Fig. 10 shows the destinations of robots from existing parabola to the target parabola. Notice that the destinations are in the segment of the parabola, where the latus rectum intersects the parabola. The latus rectum is denoted with a dotted line in the figure.

*c) Symmetric Type I Configuration:* Similar to the previous case, we also select the parabola with the larger latus rectum as the target pattern. If there are $k$ robots on the line of symmetry, then we have $(n+k)/2$ uniform points on each side of the pattern. The robots follow the local ordering for each side and set their destination to a point on the same side

[5]The latus rectum is the line that passes through the focus of the parabola and parallel to the directrix. A common form of representing a parabola is $x^2 = 2ly$, where $2l$ is the length of the latus rectum.

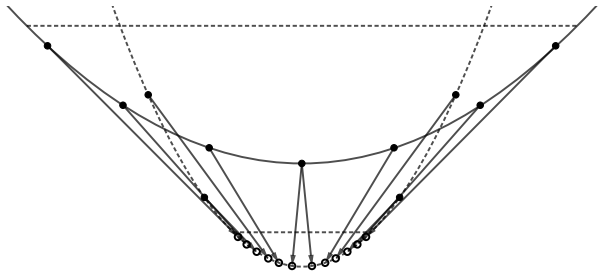

Fig. 11. A symmetric Type I configuration for $f = 4$

of the line of symmetry. The robots on the line of symmetry have two potential destinations and choose one according to their local orientation. The uniform distance $c = \mathcal{P}/(n+k)$, where $\mathcal{P}$ is the length of arc of parabola between the latus rectum intersections. Fig. 11 shows the destinations of the robots as circle on the target parabola (dotted). Observe that there was only one robot on the line of symmetry. It has two potential destinations in the resulting configuration. Similar to the previous case, we choose the uniform points between intersection points of the latus rectum with the parabola.

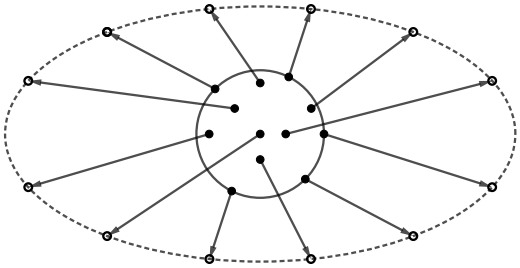

Fig. 12. Destinations for Type O configuration for $f = 5$

In the case of parabola, a Type I configuration cannot have rotational symmetry. If the robots are in a rotational symmetric configuration, then we cannot make a parabola through the robots. We discuss more in §VII, where we address the case when we need to form a circle in case of $f = 4$.

### *E. Ellipse Formation ($f = 5$)*

An ellipse is the target pattern for five faults. We have the following different configurations.

*a) Type O Configuration:* The length of the semi-major axis[6] of chosen ellipse is twice the diameter and semi-minor axis[7] is equal to the diameter with the center of the ellipse at the center of the smallest enclosing circle. The uniform points are on the ellipse with uniform distance $c = \mathcal{P}/n$, where $\mathcal{P}$ is the perimeter of the ellipse (ref. Fig. 12).

*b) Asymmetric Type I Configuration:* The ellipse passing through five points is unique. The uniform points are chosen at uniform distance $c = \mathcal{P}/n$, where $\mathcal{P}$ is the perimeter of the target ellipse. Similar to the case of circle, we choose

[6]The distance of the farthest point from the center of ellipse.

[7]The distance of the closest point from the center of ellipse.

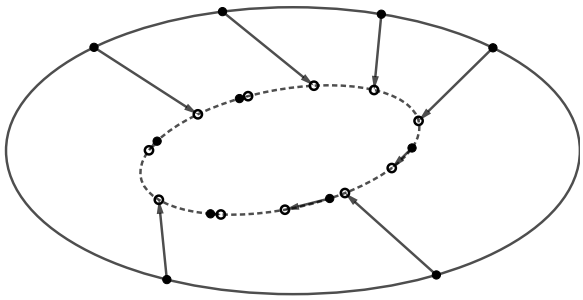

Fig. 13. Destinations for asymmetric Type I configuration for $f = 5$

the uniform points such that it does not overlap with existing points and intersection points. Fig. 13 shows the destinations of robots from a larger ellipse to a smaller ellipse on which the uniform positions lie.

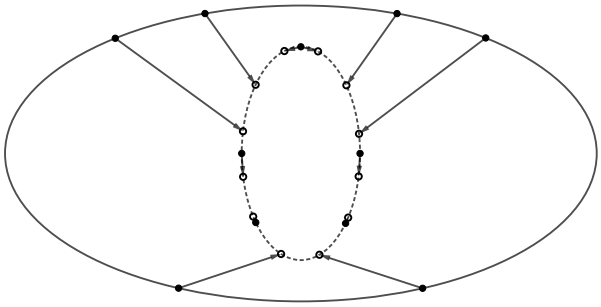

Fig. 14. Destinations for symmetric Type I configuration for $f = 5$

*c) Symmetric Type I Configuration:* Similar to the case of the parabola, the Type I configuration for the ellipse cannot have rotational symmetry. We present the strategy for a reflective symmetric configuration next. Let $k$ be the number of robots on the line of symmetry in the current configuration. The uniform distance is $c = \mathcal{P}/(n + k)$. The points are at a distance $c/2$ from the line of symmetry. Fig. 14 shows the destinations. Notice that there is a robot on the line of symmetry which has two potential destinations.

## VI. General Algorithmic Framework

The algorithm broadly has two steps based on the current configuration of the robots.

Step 1: Determine the faulty robots.
Step 2: Move to a pattern passing through the faulty robots.

Since the robots are oblivious, they cannot distinguish between the Step 1 and Step 2. Hence, the properties required for Step 1 have to be applicable to Step 2 and vice versa. For Step 1, a simple process to determine all the faulty robots in a single round is to move all the robots, such that the robots which do not move, will not lie on the pattern points. We determine the pattern points uniformly so that all pattern points are equidistant along the pattern. This helps us in determining the faulty robots since they would not lie on a pattern point. For Step 2, the pattern determined from the faulty robot positions has to be unique, so that all the robots agree on the pattern. Overall, the algorithm needs to determine a unique pattern such that all robots agree on the pattern, and all robots are required to move to achieve the pattern.

**Lemma 4.** *The destinations of all robots are distinct.*

*Proof.* We always follow the ordering to determine the destinations for each robot. In the asymmetric cases, we have the ordering, which creates a one-to-one map from the current position of a robot to its destination. In case of symmetric configurations, the robots which are present on the line of symmetry have two potential destinations (from a global perspective) and choose one of them according to their local orientation. Hence the destinations are distinct. □

Next, we prove that there are no overlapping points between the current configuration and the set of destinations. The set of destinations is the set of all potential destination points for the robots in the current configuration. Since some robots are faulty, two consecutive configurations may have those points as common points. Had the robots been non-faulty, then they would have moved to a point that is not in the current configuration.

**Lemma 5.** *Given configuration $\mathcal{C}$ and a destination set $\mathcal{C}'$, we have $\mathcal{C} \cap \mathcal{C}' = \phi$.*

*Proof.* For a Type O configuration, the destinations are outside the SEC, and thus we have no intersection. For a Type I configuration, the uniform points are chosen such that they do not overlap with the intersection points or the potentially faulty robot positions through which the target pattern is determined. Let there be $k$ points through which the target pattern is being determined for a Type I configuration. In this case, we assume that the robots are faulty, but that may not be true. So we need to make sure that the destination points do not overlap with any existing point. Otherwise, we would have two robots at a point in the resulting configuration. Let $c$ be the uniform distance at which the uniform positions to be chosen. If all $k$ points belong to the same uniform set, then we may have to choose another set of points since there is a possibility that the uniform points chosen, overlap with the intersection points of the current pattern and the target pattern. So we choose a different value of $c$ and repeat the process again. For an asymmetric Type I configuration, we choose $c' = nc/(n+1)$. In that case, we may obtain more points where we associate two destinations for the robot with the highest rank. Hence there is no intersection between the given configuration and the destination configuration. □

### A. Determining Faulty Robots

There are two types of initial configurations where we need to determine the faulty robots, i.e., arbitrary configurations and intermediate configurations. The destinations for the robots are such that no point in $\mathcal{C}_0$ overlaps with any point in $\mathcal{C}_1$. For

an arbitrary initial configuration, the target pattern is scaled such that no point in the target pattern lies on or inside the smallest enclosing circle of $\mathcal{C}_0$. Since $\mathcal{C}_0$ is asymmetric, we can always uniquely scale the pattern. We have the following two lemmas.

**Lemma 6.** *It takes one round to determine all the faulty robots for a Type O configuration for $f \in \{2, 3, 4, 5\}$.*

*Proof.* Since we need to determine all the faulty robots, we make sure that all the robots are required to move to form the target pattern. No point in the existing configuration coincides with the points in the target pattern. According to the algorithms in the previous section, we ensure that the robots move to a point outside the smallest enclosing circle of $\mathcal{C}_0$. So in $\mathcal{C}_1$, the robots which lie inside the pattern are faulty. For $f = 2$, the faulty robots would be in-between the first and the second robot of the existing line. For $f = 3$ and $f = 5$, the faulty robots would lie inside the circle or ellipse. For $f = 4$, the robots would lie on the side of the parabola with the focus. □

**Lemma 7.** *It takes one round to determine all the faulty robots for an asymmetric Type I configuration for $f \in \{2, 3, 4, 5\}$.*

*Proof.* Similar to Lemma 6, we need to ensure that all the robots move. So, the initial intermediate configuration $\mathcal{C}_0$, should not have points overlapping with the target pattern. According to the algorithms, we ensure that the target pattern points and the current configuration points do not have common points. We also ensure that the intersection points of the current configuration pattern and the target pattern are not present in the set of destination points. Even if the robots which are faulty already part of the target pattern, they do not lie on the uniform points. Hence we can figure out which robots are faulty in the resulting configuration. □

### *B. Determining the Pattern through faulty robots*

In this section, we show that a unique pattern exists that passes through all the faulty robots.

**Lemma 8.** *The target pattern passing through the faulty robots in $\mathcal{C}_1$ can be uniquely determined.*

*Proof.* The pattern is determined uniquely for a given value of $f$. For $f = 2, 3$ and 5, the line, circle, and ellipse passing through the points are unique. For $f = 4$, there can be two conjugate parabolas passing through four points. In this case, the parabola with the larger latus rectum is chosen as the target pattern. From Lemma 5, we know that the destination points do not have a common point with the points in the previous configuration. From the quasi-uniform configuration, we can determine the robots which are not present at a uniform point. Hence, we can determine the faulty robots in $\mathcal{C}_1$ and the corresponding pattern. □

### *C. Termination*

We can now show that the algorithm terminates, and we can determine the faulty robots in the terminal configuration. Observe that by combining Lemma 6, 7 and 8 we obtain:

**Theorem 9.** *Starting from any initial asymmetric configuration, the algorithm terminates in at most two rounds.*

Since the algorithm does not do anything for a terminal configuration, we cannot determine the faulty robots if the initial configuration is a terminal configuration. Moreover, in a terminal configuration, our algorithm designs yield the following distribution of robots in the plane starting from a configuration other than the terminal configuration.

**Corollary 1.** *Starting from a configuration other than the terminal configuration, the non-faulty robots are at uniform pattern points in the terminal configuration.*

*Proof.* The destinations are always at uniform points spread over the target pattern. So whenever a non-faulty robot moves, it ends up at a uniform pattern point. Note that the resulting configuration may not be uniform due to the faulty robots. The non-faulty robots occupy the uniform points in a quasi-uniform configuration. □

From Lemma 5 and Corollary 1, we have the following Corollary.

**Corollary 2.** *The faulty robots can be determined from a terminal configuration unless it is the initial configuration.*

We present a transition diagram for the configurations in our algorithm in the Fig. 15.

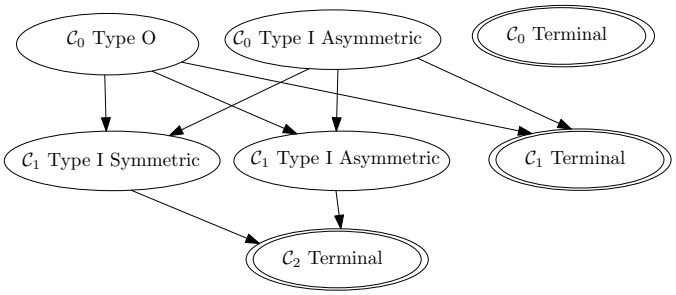


Fig. 15. Transition diagram of the configurations

## VII. Discussion

In the following, we show how to relax our model and extend the previous results in several directions. In particular, we extend the behavior of the algorithm in the absence of the assumption considered in §II. We show that with small modifications to the algorithm, we can subvert some assumptions.

### *A. At most $f$ instead of exactly $f$*

We extend the definition of Type I configuration to include the configurations where at most $f$ robots are not in the pattern in the current configuration. Since we need exactly $f$ robots to determine the target pattern, if $f' < f$ are not in the pattern, we choose a target pattern passing through those $f'$ and the first $f - f'$ robots in the ordering to determine the pattern.

### B. $\mathcal{C}_0$ with reflective symmetry

For a configuration with a single line of symmetry, we can always follow the strategies described for Type I symmetric configurations in the algorithms from §V. The robots on the line of symmetry have two destinations on either side of the line of symmetry. According to their local orientation, they choose one of the destinations.

### C. Lower order patterns for higher $f$

We add special cases if the robots are collinear (resp. co-circular) for $f \in \{3, 4, 5\}$ (resp. $f \in \{4, 5\}$). In this case, the robots form a line (resp. circle). If the initial configuration has this situation, then it is not possible to determine the faulty robots. Then the configuration in the next step becomes an arbitrary configuration. In total, we need three steps to achieve pattern formation instead of two.

## VIII. Conclusion

This paper initiated the study of distributed algorithms for pattern formation with faulty robots. In particular, we presented an algorithmic framework that allows solving many basic formation problems in two parallel rounds, which is optimal given the lower bound also presented in this paper. We regard our work as a first step and believe it opens several interesting avenues for future research. In particular, it will be interesting to study pattern formation problems for more advanced robots under failures, as well as randomized algorithms. It will also be interesting to generalize our failure model, e.g., to support transient crash faults and byzantine faults.

## Acknowledgement

The research work and the visit to the University of Vienna by the Indian authors are sponsored by Award No. ODF/2018/001055 under the Overseas Visiting Doctoral Fellowship scheme. The authors are grateful to the Science and Engineering Research Board, Department of Science and Technology, Government of India, for their support.

## References

[1] Noa Agmon and David Peleg. Fault-tolerant gathering algorithms for autonomous mobile robots. *SIAM J. Comput.*, 36(1):56–82, 2006.

[2] Cédric Auger, Zohir Bouzid, Pierre Courtieu, Sébastien Tixeuil, and Xavier Urbain. Certified impossibility results for byzantine-tolerant mobile robots. In *Stabilization, Safety, and Security of Distributed Systems - 15th Intl. Symposium, SSS 2013, Osaka, Japan, November 13-16, 2013. Proceedings*, pages 178–190, 2013.

[3] Subhash Bhagat and Krishnendu Mukhopadhyaya. Fault-tolerant gathering of semi-synchronous robots. In *Proceedings of the 18th International Conference on Distributed Computing and Networking, Hyderabad, India, January 5-7, 2017*, page 6, 2017.

[4] Zohir Bouzid, Shantanu Das, and Sébastien Tixeuil. Gathering of mobile robots tolerating multiple crash faults. In *IEEE 33rd Intl. Conference on Distributed Computing Systems, ICDCS 2013, 8-11 July, 2013, Philadelphia, Pennsylvania, USA*, pages 337–346, 2013.

[5] Quentin Bramas and Sébastien Tixeuil. Wait-free gathering without chirality. In *SIROCCO 2015, Montserrat, Spain, July 14-16, 2015, Post-Proceedings*, pages 313–327, 2015.

[6] Quentin Bramas and Sébastien Tixeuil. Brief announcement: Probabilistic asynchronous arbitrary pattern formation. In *Proceedings of the 2016 ACM Symposium on Principles of Distributed Computing, PODC 2016, Chicago, IL, USA, July 25-28, 2016*, pages 443–445, 2016.

[7] Scott Camazine, Jean-Louis Deneubourg, Nigel R Franks, James Sneyd, Eric Bonabeau, and Guy Theraula. *Self-organization in biological systems*. Princeton university press, 2003.

[8] Sruti Gan Chaudhuri and Krishnendu Mukhopadhyaya. Gathering asynchronous transparent fat robots. In *Distributed Computing and Internet Technology, 6th International Conference, ICDCIT 2010, Bhubaneswar, India, February 15-17, 2010. Proceedings*, pages 170–175, 2010.

[9] Sruti Gan Chaudhuri and Krishnendu Mukhopadhyaya. Leader election and gathering for asynchronous fat robots without common chirality. *J. Discrete Algorithms*, 33:171–192, 2015.

[10] Mark Cieliebak and Giuseppe Prencipe. Gathering autonomous mobile robots. In *SIROCCO 9, Proceedings of the 9th Intl. Colloquium on Structural Information and Communication Complexity, Andros, Greece, June 10-12, 2002*, pages 57–72, 2002.

[11] Reuven Cohen and David Peleg. Robot convergence via center-of-gravity algorithms. In *Proc. SIROCCO*, pages 79–88, 2004.

[12] Reuven Cohen and David Peleg. Convergence properties of the gravitational algorithm in asynchronous robot systems. *SIAM J. Comput.*, 34(6):1516–1528, June 2005.

[13] Shantanu Das, Paola Flocchini, Nicola Santoro, and Masafumi Yamashita. Forming sequences of geometric patterns with oblivious mobile robots. *Distributed Computing*, 28(2):131–145, 2015.

[14] Xavier Défago, Maria Gradinariu Potop-Butucaru, Julien Clément, Stéphane Messika, and Philippe Raipin Parvédy. Fault and byzantine tolerant self-stabilizing mobile robots gathering - feasibility study -. *CoRR*, abs/1602.05546, 2016.

[15] Paola Flocchini, Giuseppe Prencipe, and Nicola Santoro. *Distributed Computing by Oblivious Mobile Robots*. Synthesis Lectures on Distributed Computing Theory. Morgan & Claypool Publishers, 2012.

[16] Paola Flocchini, Giuseppe Prencipe, Nicola Santoro, and Peter Widmayer. Hard tasks for weak robots: The role of common knowledge in pattern formation by autonomous mobile robots. In *Proc. ISAAC*, pages 93–102, 1999.

[17] Paola Flocchini, Giuseppe Prencipe, Nicola Santoro, and Peter Widmayer. Gathering of asynchronous oblivious robots with limited visibility. In *STACS 2001, 18th Annual Symposium on Theoretical Aspects of Computer Science, Dresden, Germany, February 15-17, 2001, Proceedings*, pages 247–258, 2001.

[18] Nao Fujinaga, Yukiko Yamauchi, Hirotaka Ono, Shuji Kijima, and Masafumi Yamashita. Pattern formation by oblivious asynchronous mobile robots. *SIAM J. Comput.*, 44(3):740–785, 2015.

[19] Debasish Pattanayak, Kaushik Mondal, H. Ramesh, and Partha Sarathi Mandal. Gathering of mobile robots with weak multiplicity detection in presence of crash-faults. *J. Parallel Distrib. Comput.*, 123:145–155, 2019.

[20] Giuseppe Prencipe. Impossibility of gathering by a set of autonomous mobile robots. *Theor. Comput. Sci.*, 384(2-3):222–231, October 2007.

[21] Ichiro Suzuki and Masafumi Yamashita. Distributed anonymous mobile robots: Formation of geometric patterns. *SIAM J. Comput.*, 28(4):1347–1363, 1999.

[22] Yusaku Tomita, Yukiko Yamauchi, Shuji Kijima, and Masafumi Yamashita. Plane formation by synchronous mobile robots without chirality. In *21st International Conference on Principles of Distributed Systems, OPODIS 2017, Lisbon, Portugal, December 18-20, 2017*, pages 13:1–13:17, 2017.

[23] Masafumi Yamashita and Ichiro Suzuki. Characterizing geometric patterns formable by oblivious anonymous mobile robots. *Theor. Comput. Sci.*, 411(26-28):2433–2453, 2010.

[24] Yukiko Yamauchi, Taichi Uehara, Shuji Kijima, and Masafumi Yamashita. Plane formation by synchronous mobile robots in the three-dimensional euclidean space. *J. ACM*, 64(3):16:1–16:43, 2017.

[25] Yukiko Yamauchi and Masafumi Yamashita. Pattern formation by mobile robots with limited visibility. In *Structural Information and Communication Complexity - 20th International Colloquium, SIROCCO 2013, Ischia, Italy, July 1-3, 2013, Revised Selected Papers*, pages 201–212, 2013.