Distributed Algorithm for Circle Formation of Disoriented Mobile Robots

Xavier Défago Akihiko Konagaya

January 27, 2003
KS-RR-2003-001

School of Knowledge Science
Japan Advanced Institute of Science and Technology
Asahidai 1–1, Tatsunokuchi
Nomi, Ishikawa, 923-1292, JAPAN
defago@jaist.ac.jp / kona@jaist.ac.jp

ISSN 1347-1570

Distributed Algorithm for Circle Formation of Disoriented Mobile Robots

Xavier DÉFAGO and Akihiko KONAGAYA

Graduate School of Knowledge Science
Japan Advanced Institute of Science and Technology (JAIST)
1-1 Asahidai, Tatsunokuchi, Nomigun, Ishikawa 923-1292, Japan

Email: defago@jaist.ac.jp, kona@jaist.ac.jp

Abstract This paper proposes a distributed algorithm by which a collection of mobile robots roaming on a plane move to form a circle. The algorithm operates under the premises that robots (1) are unable to recall past actions and observations (i.e., oblivious), (2) cannot be distinguished from each others (i.e., anonymous), (3) share no common sense of direction, and (4) are unable to communicate in any other ways than by observing each others position.

Keywords Distributed algorithms, Cooperative mobile robots, Pattern formation, Self-stabilization, Computational geometry

1 Introduction

Mobile computing systems, devices, and applications are gradually becoming more and more pervasive, while the theoretical foundations are still not yet fully established. Current researches addressing the principles of mobile computing are mostly aimed at systems in which mobility occurs as an external factor, such as mobile ad hoc networking, mobile information systems, or ubiquitous computing. Our research however addresses systems for which the mobility must be controlled. More specifically, we consider the family of cooperation problems among groups of autonomous mobile robots, such as autonomous transport vehicles (ATV), micro-electromechanical systems (MEMS), or nanomachines [5]. We believe that both aspects of mobile computing must be studied conjointly, and we hope, in the long run, to establish the link between mobile networking and mobile robotics.

In this paper, we consider a system in which robots are modeled as points that move on the plane. The robots have no identity, no memory of past actions, no common sense

An earlier version of this paper appeared in *Proc. 2nd ACM Annual Workshop on Principles of Mobile Computing (POMC'02)*. This report refines some of the constraints and corrects a couple of unfortunate mistakes.

of distance and direction, and execute the same deterministic algorithm. Besides, robots are unable to communicate directly, and can only interact by observing each others position. In this model, we address the problem which consists in making the robots form a circle, for which we give a self-stabilizing distributed algorithm. This problem is important because, forming a circle provides a way for robots to agree on both a common origin and a common unit distance [14]. From a more practical standpoint, pattern formation problems provide a first step toward flocking, i.e., allowing a group to move in formation [10]. The formation of geometrical patterns and flocking are both useful, for instance, for the self-positioning of mobile base stations in a mobile ad hoc network environment, e.g., as considered by Chatziagiannakis et al. [3].

Our principal motivation for studying the problem of circle formation is however different. We indeed intend to use this problem as a starting point for studying the role and strengths of several different communication models. For instance, the algorithm presented in this paper relies exclusively on the fact that robots can detect each others position, as is the case with vision (with unlimited range). It is now interesting to see whether replacing vision with other communication models (e.g., ad hoc networking with directional antennas) still allows for solving the circle formation problem. This question is however not addressed here and left for later studies.

As already mentioned, the main contribution of this paper is to propose a distributed algorithm by which a group of mobile robots eventually form a circle. The algorithm deterministically forms a circle within finite time, and asymptotically converges toward a situation wherein all robots are uniformly distributed on the boundary of that circle. Beside, the algorithm does not require robots to memorize past actions, and hence it is self-stabilizing,[1] provided that no two robots are exactly identical (i.e., having both the same initial position and the same local coordinate system).[2]

Related work A vast amount of researches exist in the context of cooperative mobile robotics. Most of it uses diverse heuristics such as free market optimization (e.g., [6]) or swarm intelligence (e.g., [11]). However, only few studies take the problem from a computational standpoint. This can be partly explained by the difficulty of the task, and the fact that heuristics are perceived as a way to circumvent that difficulty. Debest [4] briefly discusses the formation of a circle by a group of mobile robots as an illustration of self-stabilizing distributed algorithms. He discusses the problem, but does not really provide an algorithm. Sugihara and Suzuki [13] propose several algorithms for the formation of various geometrical patterns. They propose an algorithm for the formation of an approximation of a circle, based on heuristics. In some cases, the shape obtained with their algorithm is a Reuleaux triangle (a hybrid shape, between a triangle and a circle) rather than a circle. Suzuki and Yamashita [15] propose a non-oblivious

[1] Self-stabilization is the property of a system which, started in an arbitrary state, always converges toward a desired behavior [7].

[2] In fact, the first part of the algorithm is self-stabilizing without this assumption, but the second part might not always converge if two robots with the same coordinate system happen to have the same initial position. This is because it would be impossible to separate them in a deterministic manner.

algorithm for the formation of a regular polygon. In other words, the robots eventually reach a configuration in which they are arranged at regular intervals on the boundary of a circle. To achieve this, they must however require the robots to be able to remember all past actions. Our paper actually builds upon their work and, most notably, inherits from the definition of their model. Under the same model, Ando et al. [1] propose an algorithm by which robots with a limited range of vision gather to a point. Flocchini et al. [9] give an algorithm to solve the same problem in a slightly different model; dropping the assumption of instantaneous computation and movement, but assuming a common sense of direction as given by compasses. Flocchini et al. [8] study the solvability of the formation of arbitrary patterns, depending on how much common knowledge the robots initially have about a global coordinate system. Uny Cao et al. [16] provide a wide survey of researches in cooperative mobile robotics, and observe that only few researches take the problem from a computational point of view. This observation is later echoed by Flocchini et al. [9].

Structure The rest of the paper is structured as follows. Section 2 presents the system model, the definition of the problem, and the notation. Section 3 gives an intuition of the algorithm and a decomposition into two subproblems: the formation of a circle (detailed in Sect. 4) and the convergence toward a uniform distribution (detailed in Sect. 5). Finally, Section 6 discusses future directions and several possible optimizations.

2 System Model and Definitions

2.1 System Model

The system model considered in this paper, defined by Suzuki and Yamashita [15], considers a collection of anonymous mobile robots evolving asynchronously on the Euclidean plane, with no common origin, unit distance, or sense of direction. More precisely, the model is defined as follows.

Each robot r_i is modeled as a mobile processor with infinite memory, and a sensor to detect the instantaneous position of all robots. The robots move in an unbounded two dimensional space devoid of any landmark. Each robot r_i uses its own local x-y coordinate system (origin, orientation, distance) and has no particular knowledge of the local coordinate system of other robots, nor of a global coordinate system. It is assumed that robots are infinitely small (i.e., points), never collide, and that two or more robots may simultaneously occupy the same physical location. Further, it is assumed that robots can observe, compute, and move with infinite decimal precision.

Robots are anonymous in the sense that they are unable to uniquely identify themselves, neither with a unique identification number nor with some external distinctive mark (e.g., color, flag). Besides, all robots execute the same deterministic algorithm,[3]

[3]By deterministic, we mean that any two independent executions of the algorithm with identical input values always yield the same output. In particular, this rules out the use of randomization.

and thus have no way to generate a unique identity for themselves.

Time is represented as an infinite sequence of time instants $t_0, t_1, t_2, \ldots$, during which each robots can be either *active* or *inactive*. Each time instant during which a robot becomes active, the robot computes a new position using a given algorithm, and moves toward that position. Conversely, when a robot is inactive, it stays still. It is further assumed that initially all robots occupy a different position.

The activation of robots is determined by an activation schedule, unpredictable and unknown to robots. At each time instant, a subset of the robots become active, with the guarantees that (1) every robot becomes active at infinitely many time instants, and (2) at least one robot is active during each time instant.[4] As a special case, robots are said to be synchronized when all of them become active at every time instant.

In this model, the algorithm consists of a deterministic function φ that is executed by every robot r_i each time it becomes active. The arguments to φ consist of the current position of the robot, and a multiset of points containing the observed position of all robots at the corresponding time instant. All positions are of course expressed in terms of the local coordinate system of r_i. The value returned by φ is the new position for r_i, which must be within one distance unit of the previous position, as measured by r_i's own coordinate system. For simplicity, it is assumed that obtaining the information about the system, computing the new position, and moving toward it is instantaneous. Note that arguments to φ include only current information and thus the algorithm can use no knowledge of the past. We hence say that the robots are *oblivious* because they are unable to remember any past actions or observations. It can be argued [15] that, as a result, the algorithm defined by φ is self-stabilizing. This is however not necessarily true, as this largely depends on the exact definition of self-stabilization.

2.2 Activation Schedule and Livelock

In the Suzuki-Yamashita model [15], it is difficult for the robots to coordinate their actions. This is largely because of of the dilemma caused by the unpredictability of the activation schedule. Let us illustrate this point by an example.

In a system with two robots r_1 and r_2 that occupy distinct positions initially, consider the problem of having them eventually move to the same location. We can now see the dilemma that r_1 faces (also r_2 by symmetry) by considering the naive solution that follows. When it is activated, robot r_1, assuming that the other robot is inactive, moves directly to the position occupied by r_2. In this case, the problem is solved in one step if r_2 indeed remains stationary. However, if r_2 happens to be active simultaneously, it takes the same action as r_1 and hence moves to occupy the position that r_1 has just left. Consequently, if the activation schedule is such that the two robots are always activated at the same time, the system remains caught in a livelock with the two robots swapping positions endlessly.

[4] As the duration of the interval between two time instants is by no means fixed, the second condition incurs no loss of generality. It is in fact only required for convenience.

2.3 Problem Definition

The problem addressed in this paper is the formation of a circle by a set of mobile robots. We define the circle formation problem as follows.

Problem 2.1 (UNIFORM CIRCLE FORMATION)
Given a group of n robots $r_1, r_2, \ldots, r_n$ with distinct positions and located arbitrarily on the plane, eventually arrange them at regular intervals on the boundary of a non degenerate circle (i.e., with finite radius greater than zero).

In the remainder of this paper, we decompose Problem 2.1 into two subproblems. The first subproblem (Prob. 2.2) is a weaker version of Problem 2.1, wherein a circle is formed without the requirement for uniform intervals between consecutive robots. The second subproblem (Prob. 2.3) consists in transforming a configuration in which robots are arranged arbitrarily on the boundary of a circle, into one where the robots are arranged uniformly.

Problem 2.2 (CIRCLE FORMATION)
Given a group of n robots $r_1, r_2, \ldots, r_n$ with distinct positions and located arbitrarily on the plane, arrange them to eventually form a non degenerate circle.

Problem 2.3 (UNIFORM TRANSFORMATION)
Given that a group of n robots $r_1, r_2, \ldots, r_n$ with distinct positions are located on the boundary of a non degenerate circle, eventually arrange them at regular intervals on the boundary of the circle.

2.4 Definitions and Notation

Position Given a robot r_i, $p_i(t)$ denotes its position at time t, according to some global x-y coordinate system, and $p_i(0)$ is its initial position. $P(t) = \{p_i(t) \| 1 \leq i \leq n\}$ denotes the multiset of the positions of all robots at time t. When this is not ambiguous, we sometimes mention a robot, implicitly referring to its position rather than the robot itself.

Voronoi diagram The *Voronoi diagram* $\mathrm{Voronoi}(P)$, also called Dirichlet tesselation, of a set of points $P = \{p_1, p_2, \ldots, p_n\}$ is a subdivision of the plane into n cells, one for each point in P. The cells have the property that a point q belongs to the Voronoi cell of point p_i, denoted $\mathrm{Vcell}_{p_i}(P)$, if and only if, for any other point $p_j \in P$, $\mathrm{dist}(q, p_i) < \mathrm{dist}(q, p_j)$, where $\mathrm{dist}(p, q)$ is the Euclidean distance between p and q. In particular, the strict inequality means that points located on the boundary of the Voronoi diagram do not belong to any Voronoi cell. A Voronoi diagram is for instance depicted on Figure 1(a) (Sect. 3.2). Significantly more details about Voronoi diagrams and their principal applications are surveyed by Aurenhammer [2].

Smallest enclosing circle The *smallest enclosing circle* of a set of points P is denoted by $\mathrm{SEC}(P)$. It can be defined by either two opposite points, or by at least three points. The smallest enclosing circle is unique, and can be computed in $O(n \log n)$ (e.g., [12]).

3 Algorithm Intuition

Given the Suzuki-Yamashita model [15] (see Sect. 2.1) and an initial configuration where a collection of robots are located arbitrarily on the plane, the algorithm ensures that the system converges toward a configuration of robots that is a valid solution to the Uniform Circle Formation problem (Prob. 2.1).

Informally, the algorithm relies on the fact that the environment observed by all robots is the same, in spite of the difference in local coordinate system. Besides, the smallest enclosing circle is unique and depends only on the relative positions of the robots. So, the algorithm makes sure that the smallest enclosing circle remains invariant and uses it as a common reference. The invariance is ensured by self-imposing several constraints on the movement of robots (Sect. 3.2).

The presentation of the algorithm is separated into two distinct parts, each of which solves a distinct subproblem. The first subproblem consists in placing the robots along the boundary of a circle, without considering their relative positions. The second subproblem consists in starting from there, and arranging robots evenly along the boundary of the circle. Section 3.1 describes this separation and explains how the parts can be reconciled.

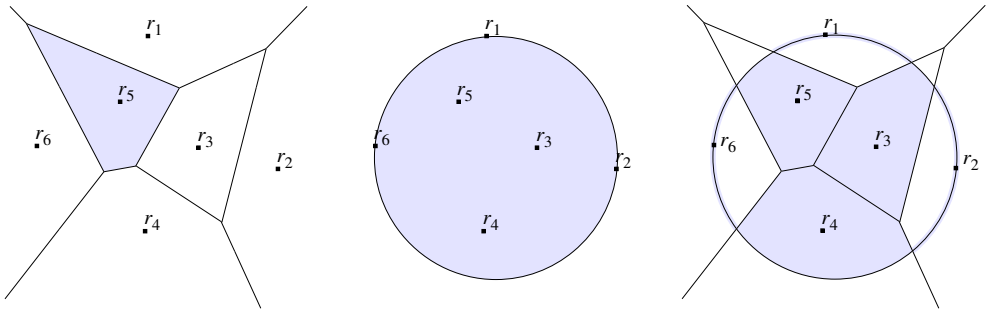
3.1 Separation

As mentioned earlier, the algorithm is decomposed into solving two different subproblems, each of which is solved by a different algorithm. More specifically, φ_{circle} is the algorithm that solves Prob. 2.2 (Circle Formation), and $\varphi_{uniform}$ the one to solve Prob. 2.3 (Uniform Transformation). Algorithm 1 combines those two algorithms into one that solves Prob. 2.1 (Uniform Circle Formation).

Algorithm 1 Combining the parts

function $\varphi(P, p_i)$

1: **if** p_i is in the interior of $\mathrm{SEC}(P)$ **then**
2: $\{$*Algorithm 2*$\}$
3: $\varphi_{circle}(P, p_i)$
4: **else**
5: $\{$*Algorithm 3*$\}$
6: $\varphi_{uniform}(\{p \in P | p \in \mathrm{SEC}(P)\}, p_i)$
7: **end if**



(a) Restr. 1: The movement of r_5 is constrained by the interior of its Voronoi cell

(b) Restr. 2: The movement of r_5 is constrained by the smallest enclosing circle

(c) Restr. 1–4: r_1, r_2, r_6 move on the boundary of the enclosing circle (Restr. 3–4); r_3, r_4, r_5 move in the interior (Restr. 1–2)

Figure 1: Illustrating the self-imposed restrictions on movement.

3.2 Self-Imposed Restrictions on Movement

In order to solve the given problems, it is necessary to impose some restrictions on the movements of robots in addition to those inherent to the system model. Doing so ensures that (1) no two robots occupy the same position simultaneously (Restr. 1), and that (2) the smallest circle enclosing all robots remains invariant (Restr. 2–4).

Restriction 1
A robot always moves toward a point that is inside its Voronoi cell.

Restriction 2
No robot ever moves beyond the boundary of the smallest circle enclosing all robots.

Restriction 3
All robots located on the boundary of the smallest enclosing circle remain on that boundary.

Restriction 4
Robots located on the circumference of the smallest enclosing circle do not move unless there are at least three such robots with distinct positions.

Restriction 5
Let r_i be an activated robot located on the boundary of $\mathrm{SEC}(P)$. Let $next(r_i) \in \mathrm{SEC}(P)$ (resp., $prev(r_i) \in \mathrm{SEC}(P)$) denote the directmost neighbor of r_i clockwisely (resp., counterclockwisely) that is not colocated with r_i. Let θ_{prev_i} and $\theta next_i$ be the angular distance between r_i and the two neighbors. Then, Then, the angular movement

of r_i, denoted α_{r_i}, is restricted as follows:

$$\frac{\theta_{prev_i} - \Pi}{2} \leq \alpha_{r_i} \leq \frac{\Pi - \theta_{next_i}}{2}$$

Lemma 3.1
Under Restriction 1, if two robots have distinct positions at some time instant, then they always have distinct positions afterward.

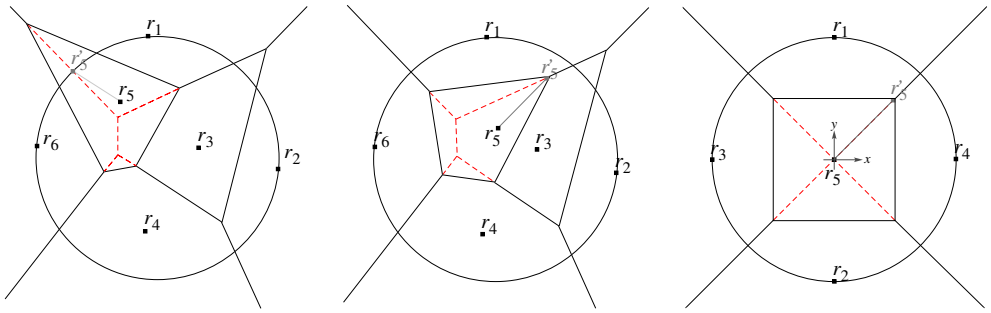
Proof *(sketch)* The proof is by induction, where the induction step directly follows from Restriction 1 and the definition of Voronoi cells. Indeed, according to the definition of Voronoi cells (see Sect. 2.4), no point on the plane can simultaneously belong to more than one cell. ■

Theorem 3.1
Under Restrictions 1–5, the smallest enclosing circle of all robots is invariant.

Proof Let $\mathrm{SEC}(t)$ and $\mathrm{SEC}(t+1)$ denote the smallest enclosing circle at time instants t and $t+1$ respectively (shortcut for $\mathrm{SEC}(P(t))$ and $\mathrm{SEC}(P(t+1))$). We prove that, regardless of the activation schedule, $\mathrm{SEC}(t)$ and $\mathrm{SEC}(t+1)$ must be identical, and the rest follows by induction.

Assume, by contradiction, that there is a time instant t for which $\mathrm{SEC}(t)$ and $\mathrm{SEC}(t+1)$ are different. First, we observe that this cannot be caused by the movement of a robot located at the interior of $\mathrm{SEC}(t)$. Indeed, such a robot could change the smallest enclosing circle only by moving outside of it, which is prevented by Restriction 2. Therefore, $\mathrm{SEC}(t+1)$ must be defined by the movement of a robot located at the boundary of $\mathrm{SEC}(t)$. There are three cases left to consider, depending on the number of robots at the boundary of $\mathrm{SEC}(t)$ or their respective position:

1. *(2 robots)* The smallest enclosing circle $\mathrm{SEC}(t)$ is defined by two robots. Those robots cannot move by Restriction 4 and hence $\mathrm{SEC}(t+1)$ remains identical to $\mathrm{SEC}(t)$.
2. *(3 robots; one quits the circle)* The smallest enclosing circle $\mathrm{SEC}(t)$ is defined by three robots, one of which moves outside the boundary of $\mathrm{SEC}(t)$. This is disallowed by Restriction 3.
3. *(3 robots; two distinct points)* The smallest enclosing circle $\mathrm{SEC}(t)$ is defined by three robots, where two of them move to the same location. This is prevented by Lemma 3.1 and the assumption that all robots have a distinct position initially.
4. *(3 robots; angular distance greater than diameter)* If the angular distance between two of the three robots is larger than the diameter, then the circle defined by the three robots and the smallest enclosing circle for the two robots are different. Since $\mathrm{SEC}(t)$ is the smallest enclosing circle at time t, the angular distance between any two of the three robots must be no greater than the diameter. By Restriction 5, the movement of two consecutive robots cannot lead them further away from each other than Π, regardless of their activation schedule.



(a) The smallest enclosing circle is reachable.

(b) The smallest enclosing circle is unreachable.

(c) Symmetry is broken by using the local coordinate system.

Figure 2: Illustration of Algorithm 2, as executed by one robot (in each case, r_5 moves toward r'_5).

When there are more than three robots on the boundary of $\mathrm{SEC}(t)$, the situation can always be reduced to one of the four cases mentioned above. It follows that $\mathrm{SEC}(t)$ and $\mathrm{SEC}(t+1)$ cannot be different; a contradiction. ■

In the sequel, when we say that an active robot r_i moves *toward* a point p, this means that r_i's next position is the point closest to p (or p itself) which satisfies all restrictions on movement described in this section, as well as the limitation on distance imposed by the model.

4 Circle Formation

The algorithm presented in this section constitutes the first part of our algorithm. It takes an arbitrary configuration in which all robots have distinct positions and, regardless of the activation schedule, eventually brings the system toward a configuration in which all robots are located on the boundary of a circle. In other words, the algorithm solves the Circle property of Section 2.3.

4.1 Algorithm

The intuition behind the algorithm is simple and we briefly describe it here (see Algorithm 2 for details). As mentioned earlier, the smallest enclosing circle is kept as an invariant. So, all robots are made to move toward the boundary of this circle.[5] So, robots that are already on the boundary do not move, and robots that are in the interior of the circle are made to move toward the boundary of the circle.

[5]The problem is trivially solved by doing nothing for cases where there are two robots or less. So, in the rest of the section, we consider the cases with three or more robots.

A robot located in the interior of the circle can find itself in either one of three types of situations. First, as illustrated for robot r_5 on Figure 2(a), the simplest situation occurs when the circle intersects the Voronoi cell of the robot. In this case, the robot selects a point on the intersection of the circle and the Voronoi cell, and moves toward it (r'_5 on Fig. 2(a)). The second situation arises when the Voronoi cell of the robot does not intersect with the circle (Fig. 2(b)). In this case, the robot selects the point in its Voronoi cell which is nearest the boundary of the circle (or farthest its center).[6] The third situation arises when, due to symmetry, there exist several such points (Fig. 2(c)). In this case, all solutions being the same, one is selected arbitrarily. This is for instance done by keeping the solution with the highest x-coordinate (and then y-coordinate) according to the local coordinate system of the robot.

Algorithm 2 Formation of an (arbitrary) circle (code executed by robot r_i)

function $\varphi_{circle}(P, p_i)$

1: **if** $p_i \in \mathrm{SEC}(P)$ **then** {r_i already on the boundary.}
2: stay still.
3: **else if** $\mathrm{Vcell}_{p_i}(P) \cap \mathrm{SEC}(P) \neq \emptyset$ **then**
4: {*c.f. Fig. 2(a)*}
5: $target := \mathrm{Vcell}_{p_i}(P) \cap \mathrm{SEC}(P) \cap \mathrm{Voronoi}(P - \{p_i\})$
6: *move toward* $target$*. (e.g.,* r'_5 *in Fig. 2(a))*
7: **else** {Voronoi cell of p_i inside circle}
8: {*c.f. Fig. 2(b)*}
9: *compute points in* $\mathrm{Vcell}_{p_i}(P)$ *closest to* $\mathrm{SEC}(P)$.
10: ***if*** *exactly one candidate exists* ***then***
11: *move toward that point. (e.g.,* r'_5 *in Fig. 2(b))*
12: ***else*** {*Several candidates exist*}
13: {*c.f. Fig. 2(c)*}
14: *select candidate with greatest* x*-coordinate, and then* y*-coordinate (i.e., first in lexical order).*
15: *move toward that point. (e.g.,* r'_5 *in Fig. 2(c))*
16: ***end if***
17: **end if**

4.2 Correctness

Lemma 4.1

Under Algorithm 2, all configurations in which the smallest enclosing circle passes through all robots are stable.

Proof This follows trivially from line 1–2 of the algorithm. ■

[6] There is the possibility that two robots in adjacent cells move to the same corner if they are activated simultaneously. However, because of the definition of Voronoi cells, they can never move to the same location (see Lemma 3.1).

Lemma 4.2
No robots moves to a position further away from $\mathrm{SEC}(P)$ (than its current position).

Proof In other words, no robots moves to a position closer to the center of the circle. Robots move only at lines 2,6,11,15. Let us consider each case for some arbitrary robot r.

1. At line 2, r stays still, so it obviously does not move toward the center.
2. At line 6, r moves from the interior of the circle toward a point located on the boundary of the circle. So, r actually progresses *away* from the center.
3. At lines 11,15, r moves to the point in its Voronoi cell that is closest to the boundary of the circle. Since a point always belongs to its own Voronoi cell, the new position must be at least as far from the center as the current one.

The robot r is unable to move away from the boundary of the smallest enclosing circle in any of the three cases, thus proving the lemma. ■

Lemma 4.3
There exists at least one robot in the interior of $\mathrm{SEC}(P)$ that can progress a non null distance toward $\mathrm{SEC}(P)$.

Proof Let us denote by P_{in} the set of all robots located in the interior of the smallest enclosing circle, and assume by contradiction that there exist a configuration in which no robot in P_{in} can progress a non null distance toward $\mathrm{SEC}(P)$. Consider now a robot r in P_{in} such that r is as far as possible from the center of $\mathrm{SEC}(P)$. By assumption, r cannot progress toward $\mathrm{SEC}(P)$.

If r cannot progress toward $\mathrm{SEC}(P)$, it must be prevented to do so by some other robot in P. Indeed, r is the robot in P_{in} farthest from the center of $\mathrm{SEC}(P)$, and so it belongs to the convex hull of P_{in}. It results that no robot in P_{in} can prevent it from reaching $\mathrm{SEC}(P)$. This is because, if we consider the Voronoi diagram of P_{in}, the Voronoi cell of r extends toward infinity (a direct consequence of r belonging to convex hull of P_{in}).

Hence, r must be blocked by some points on $\mathrm{SEC}(P)$. To block r, there must be at least two robots on $\mathrm{SEC}(P)$, say r_a and r_b. Indeed, with only one single robot on $\mathrm{SEC}(P)$ to block its way, r will invariably follow a path around the blocking point, thus avoiding it altogether.[7] The robot r and the two robots blocking its way r_a and r_b are illustrated on Figure 3. In the figure, r is represented as being located on the bisector of r_a and r_b. It is easy to see that, even if r is not originally on the bisector, it will eventually move toward it as a result of the algorithm (the location selected at line 9 is always near one of the vertices of the Voronoi cell).

Now, we consider the reachability $x_r(t)$, which is the maximal distance that r can move toward the midpoint of r_a and r_b during activation step t.[8] By assumption, there

[7] This part of the proof is omitted here due to space limitations, but will appear in an extended version of this paper.

[8] Without loss of generality, we chose to ignore here the limitation imposed by the model as it does not influence the argumentation.

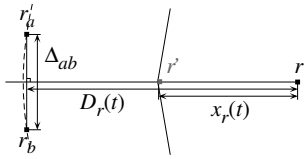


Figure 3: Progression of a robot toward the boundary of the smallest enclosing circle, when blocked by two robots.

must be a situation for which this value is zero (it can never be negative because of Lemma 4.2). So, computing $x_r(t)$ yields the following formula:[9]

$$x_r(t) = \frac{D_r(t)^2 + \left(\frac{\Delta_{ab}}{2}\right)^2}{2D_r(t)} \tag{1}$$

where $D_r(t)$ is the distance between r and the segment $\overline{r_a r_b}$ at activation step t, and Δ_{ab} is the distance between r_a and r_b. It turns out that, following Equation 1, $x_r(t)$ has the following lower bound when $D_r(t)$ varies,

$$\forall D_r(t), \quad x_r(t) \geq \frac{\Delta_{ab}}{2} > 0 \tag{2}$$

It results that the robot r can always reach a location that is strictly closer to the boundary of $\mathrm{SEC}(P)$, thus contradicting the assumption that r is unable to progress toward $\mathrm{SEC}(P)$. This hence proves the lemma. ■

Lemma 4.4
All robots located in the interior of $\mathrm{SEC}(P)$ reach its boundary after a finite number of activation steps.

Proof By Lemma 4.2 no robot ever moves backward. By Lemma 4.3, there is at least one robot r in P_{in} that can reach $\mathrm{SEC}(P)$ in a finite number of steps. Once r has reached $\mathrm{SEC}(P)$, it does not belong to P_{in} anymore. So, by Lemma 4.3, there must be another robot in P_{in} that can reach $\mathrm{SEC}(P)$ in a finite number of steps. Since there is a finite number of robots in P, there exists some finite time after which all robots are located on the boundary of the smallest enclosing circle $\mathrm{SEC}(P)$. ■

Theorem 4.1
Algorithm 2 solves the problem of Circle Formation (Prob. 2.2).

Proof By Lemma 4.4, there is a time after which all robots are located on the smallest enclosing circle, and this configuration is stable (Lemma 4.1). Consequently, Algorithm 2 solves the Circle Formation problem. ■

[9] The development of the formula is not especially difficult, and so we have decided to omit it here.

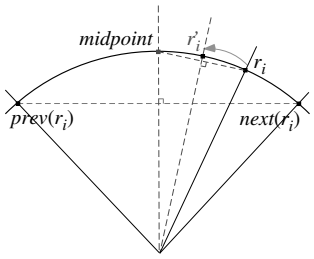


Figure 4: Robot r_i moves halfway toward the midpoint (Algo. 3).

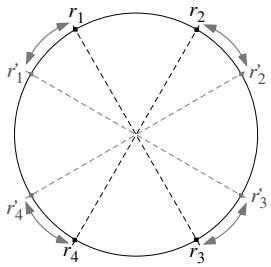


Figure 5: Moving to the midpoint can result in oscillations.

5 Uniform Transformation

Given that all robots are located on the circumference of a circle, the problem consists in distributing then evenly along that circumference. The algorithm that we propose in this section converges toward a homogeneous distribution of robots, but it does not terminate deterministically.

5.1 Algorithm

Algorithm 3 is very simple.[10] As illustrated on Figure 4, whenever a robot r_i becomes active, it considers its two direct neighbors $prev(r_i)$ and $next(r_i)$, and computes the midpoint between them, $midpoint$. Then, r_i moves halfway toward $midpoint$, as permitted by the restrictions on the movement of robots, and its own reachability radius.

Algorithm 3 Convergence toward a uniform configuration

function $\varphi_{uniform}(P, p_i)$

1: {*Assumes all robots are on the boundary of* $\mathrm{SEC}(P)$.}
2: $prev(p_i) :=$ *direct neighbor of* p_i *counterclockwise.*
3: $next(p_i) :=$ *direct neighbor of* p_i *clockwise.*
4: $midpoint :=$ *midpoint of arc* $prev(p_i), p_i, next(p_i)$.
5: $target :=$ *midpoint of arc* $midpoint, p_i$.
6: *move toward* $target$

The reason for moving halfway toward the midpoint rather than toward the midpoint itself is to prevent situations such as the one illustrated in Figure 5, where the system oscillates endlessly between two different configurations if robots are perfectly

[10] We have also designed a smarter algorithm with a better convergence rate. We will however keep its presentation for an extended version of this paper, since time and space constraints do not allow us to properly develop its description, proofs, and convergence analysis.

synchronized. The system would get stuck into an infinite cycle, and hence be unable to progress toward an acceptable solution.

5.2 Correctness

Lemma 5.1
Under Algorithm 3, all configurations in which all robots are uniformly distributed over a circle are stable.

Proof Consider an arbitrary configuration in which n robots are uniformly distributed over a circle. The angular distance between any two consecutive robots must hence be $\frac{2\Pi}{n}$. Assume that such a configuration is reached at time t. We now show that, under Algorithm 3, the system remains in the same configuration at time $t+1$, regardless of the activation schedule. Consider a robot r_i activated at time t. Since the distance between r_i and either neighbors is the same, it is easy to see that the destination computed by r_i is p_i. Thus, r_i remains in the same position. It results that no robot moves, whether activated or not. Hence the configuration is left unchanged and the lemma follows. ■

Theorem 5.1
Algorithm 3 converges toward a configuration wherein all robots are arranged at regular intervals on the boundary of a circle.

Proof *(sketch)* By Lemma 5.1, we know that any configuration where all robots are uniformly distributed over the circle is stable. In other words, the angular distance between a robot and its two direct neighbors is identical for all robots, that is, $\frac{2\Pi}{n}$.

Second, we consider $\Theta_{max}(t)$ (resp., $\Theta_{min}(t)$) the maximal (resp., minimal) angular distance between two direct neighbors at time t. The proof consists in showing that $\Theta_{max}(t)$ is monotonic decreasing, whereas $\Theta_{min}(t)$ is monotonic increasing. ■

6 Discussion & Future Work

In this paper, we have presented a distributed algorithm by which oblivious mobile robots move to form a circle. In this section, we briefly discuss possible improvements to the algorithm, as well as some important research issues for the future.

Transformation in finite time The algorithm presented in this paper can form a (non uniform) circle in finite time, but only converges toward an even distribution. We conjecture that an oblivious deterministic solution to the second subproblem (Transformation) does not exist in the Suzuki-Yamashita model for any number of robots, because of the unpredictability of the activation schedule. We believe that non oblivious solutions should exist, based on algorithms for the formation of regular polygons [15].

Weaker models The model considered in this paper [15] is actually stronger than it seems, and interesting future work will investigate the problem in weaker models. The first issue is the implicit synchrony of the model. Indeed, at each activation, a robot consecutively observes, computes, and moves as one single atomic operation. This introduces some synchrony between the activation schedule of robots. Flocchini et al. [8] define a weaker model wherein the three actions no longer needs to be atomic. A second issue is the limitless range of vision. Constraining the range, as done by Flocchini et al. [9], would make it impossible to determine the smallest enclosing circle, or count the total number of robots.

Optimizations For the sake of clarity, we have tried to keep the expression of the algorithms as simple as possible. There are however several straightforward optimizations that can be easily applied to the algorithms to improve performance, or reduce computation.

For instance, computing the smallest enclosing circle can be somewhat alleviated if the robots happen to have the ability of memorizing past actions in spite the assumptions. The circle is invariant in the absence of external influence, and so needs to be computed only once, and just verified later on.

Another example would be to combine the two parts of the algorithm in a better way. Robots could for instance execute the two parts of the algorithm simultaneously and, in the second part, use their projection on the circle, for robots that are still in the interior.

Communication models Building on the algorithm and its limitations, we intend to investigate the power of different augmentations of the model with respect to solving this problem. For instance, we are currently pursuing a study which consists in augmenting the system model with the ability to communicate, using several different communication models.

Complexity The overall computational complexity for a single robot is $O(n \log n)$ for computing the smallest enclosing circle as well as the Voronoi diagram. We did not determine the space complexity of the algorithm, but we believe it to be linear (due to the Voronoi diagrams). We intend to investigate these issues more rigorously in the future.

Convergence Although we prove that the algorithm is correct and thus eventually converges toward an acceptable solution for the circle formation problem, it would be interesting to study the convergence quantitatively. To do this, we will try to calculate an upper bound for the convergence of the first part of the algorithm. We will also try to study the convergence of the overall algorithm by using simulations.

Practical issues Our prime motivation for developing the circle formation algorithm presented in this paper is admittedly theoretical rather than practical. Although we believe that the algorithm could actually be used in practice, there are several important issues that must be addressed first. For instance, the hidden synchrony introduced by the atomicity of activation cycles is a severe practical limitation. In addition, the unlimited visibility, the infinite precision and accuracy of sensors, as well as the dimensionless nature of the robots are important points that must be addressed before the algorithm or similar solutions can be applicable to practical situations. Engineering issues constitute a very important topic that we leave for future work.

Acknowledgements

We are especially grateful to Michail Markou and Sayan Mitra for their helpful comments.

References

[1] H. Ando, Y. Oasa, I. Suzuki, and M. Yamashita. Distributed memoryless point convergence algorithm for mobile robots with limited visibility. *IEEE Trans. on Robotics and Automation*, 15(5):818–828, Oct. 1999.

[2] F. Aurenhammer. Voronoi diagrams–a survey of a fundamental geometric data structure. *ACM Comput. Surv.*, 23(3):345–405, Sept. 1991.

[3] I. Chatzigiannakis, S. Nikoletseas, and P. Spirakis. On the average and worst-case efficiency of some new distributed communication and control algorithms for ad-hoc mobile networks. In *Proc. 1st ACM Int'l Workshop on Principles of Mobile Computing (POMC'01)*, pages 1–19, Newport, RI, USA, Aug. 2001.

[4] X. A. Debest. Remark about self-stabilizing systems. *Commun. ACM*, 38(2):115–117, Feb. 1995. Technical correspondence.

[5] X. Défago. Distributed computing on the move: From mobile computing to cooperative robotics and nanorobotics (a position paper). In *Proc. 1st ACM Int'l Workshop on Principles of Mobile Computing (POMC'01)*, pages 49–55, Newport, RI, USA, Aug. 2001.

[6] M. B. Dias and A. Stentz. A free market architecture for distributed control of a multirobot system. In *Proc. of the 6th Int'l Conf. on Intelligent Autonomous Systems (IAS-6)*, pages 115–122, Venice, Italy, July 2000.

[7] S. Dolev. *Self-Stabilization*. MIT Press, 2000.

[8] P. Flocchini, G. Prencipe, N. Santoro, and P. Widmayer. Hard tasks for weak robots: The role of common knowledge in pattern formation by autonomous mobile robots. In *Proc. 10th Int'l Symp. on Algorithms and Computation (ISAAC'99)*, volume 1741 of *LNCS*, pages 93–102, Chennai, India, Dec. 1999. Springer.

[9] P. Flocchini, G. Prencipe, N. Santoro, and P. Widmayer. Gathering of asynchronous oblivious robots with limited visibility. In *Proc. 18th Annual Symp. on Theoretical Aspects of Computer Science (STACS 2001)*, volume 2010 of *LNCS*, pages 247–258, Dresden, Germany, Feb. 2001. Springer.

[10] V. Gervasi and G. Prencipe. Flocking by a set of autonomous mobile robots. Technical Report TR-01-24, Dipartimento di Informatica, Università di Pisa, Italy, Oct. 2001.

[11] M. J. B. Krieger, J.-B. Billeter, and L. Keller. Ant-like task allocation and recruitment in cooperative robots. *Nature*, 406(6799):992–995, Aug. 2000.

[12] S. Skyum. A simple algorithm for computing the smallest enclosing circle. *Information Processing Letters*, 37(3):121–125, 1991.

[13] K. Sugihara and I. Suzuki. Distributed algorithms for formation of geometric patterns with many mobile robots. *Journal on Robotics Systems*, 3(13):127–139, Mar. 1996.

[14] I. Suzuki and M. Yamashita. Agreement on a common x-y coordinate system by a group of mobile robots. In *Proc. Dagstuhl Seminar on Modeling and Planning for Sensor-Based Intelligent Robots*, Dagstuhl, Germany, Sept. 1996.

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

[16] Y. Uny Cao, A. S. Fukunaga, and A. B. Kahng. Cooperative mobile robotics: Antecedents and directions. *Autonomous Robots*, (4):1–23, 1997.