A Tight Runtime Bound for Synchronous Gathering of Autonomous Robots with Limited Visibility*

Bastian Degener
Heinz Nixdorf Institute & Department of Computer Science
University of Paderborn
bastian.degener@upb.de

Barbara Kempkes
Heinz Nixdorf Institute & Department of Computer Science
University of Paderborn
barbaras@upb.de

Tobias Langner
Computer Engineering and Networks Lab (TIK)
ETH Zurich
langnert@tik.ee.ethz.ch

Friedhelm Meyer auf der Heide
Heinz Nixdorf Institute & Department of Computer Science
University of Paderborn
fmadh@upb.de

Peter Pietrzyk
Heinz Nixdorf Institute & Department of Computer Science
University of Paderborn
toon@upb.de

Roger Wattenhofer
Computer Engineering and Networks Lab (TIK)
ETH Zurich
wattenhofer@tik.ee.ethz.ch

ABSTRACT

The problem of gathering n autonomous robots in the Euclidean plane at one (not predefined) point is well-studied under various restrictions on the capabilities of the robots and in several time models. However, only very few runtime bounds are known. We consider the scenario of *local algorithms* in which the robots can only observe their environment within a fixed viewing range and have to base their decision where to move in the next step solely on the relative positions of the robots within their viewing range. Such local algorithms have to guarantee that the (initially connected) unit disk graph defined by the viewing range of the robots stays connected at all times.

In this paper, we focus on the synchronous setting in which all robots are activated concurrently. Ando et al. [2] presented an algorithm where a robot essentially moves to the center of the smallest enclosing circle of the robots in its viewing range and showed that this strategy performs gathering of the robots in finite time. However, no bounds on the number of rounds needed by the algorithm are known. We present a lower bound of $\Omega(n^2)$ for the number of rounds as well as a matching upper bound of $\mathcal{O}(n^2)$ and thereby obtain a tight runtime analysis of the algorithm of $\Theta(n^2)$.

*Partially supported by the EU within FP7-ICT-2007-1 under contract no. 215270 (FRONTS) and DFG-project "Smart Teams" within the SPP 1183 "Organic Computing"

SPAA'11, June 4–6, 2011, San Jose, California, USA.

Categories and Subject Descriptors

F.1.2 [**Theory of Computation**]: Modes of Computation; F.2.2 [**Analysis of Algorithms and Problem Complexity**]: Nonnumerical Algorithms and Problems—*geometrical problems and computations*; I.2.9 [**Artificial Intelligence**]: Robotics—*autonomous vehicles*; I.2.11 [**Artificial Intelligence**]: Distributed Artificial Intelligence—*multiagent systems, intelligent agents*

General Terms

Algorithms, Performance, Theory

Keywords

local algorithms, distributed algorithms, robot gathering, mobile robots, multiagent systems

1. INTRODUCTION

In the future, large groups of small and cheap mobile robots can potentially replace few and expensive robots for many tasks. Thus, there is a growing interest in figuring out which kinds of tasks can be solved by such robotic teams. For mobile robots, it is especially interesting whether they can build a given formation and which sensoric and actoric capabilities are needed to do so. Naturally, the goal is to require as few capabilities as possible in order to be able to use robots that are as cheap as possible.

In this paper we study a classic mobile network problem, the robot-gathering problem. As we discuss in more detail in the related work section, robot-gathering has received considerable attention in the past few years, and there exist various model variants. We are particularly interested in the concurrent version of the problem: We are given n robots, modeled as points in the two-dimensional Euclidean plane, and these robots want to gather at a single point. In each synchronous round, every robot observes the plane and the other robots, decides where to move, and moves there, concurrently with all other robots. The next round does not start before the last movement has finished. If robots have full visibility, the problem is trivial as all robots can compute the unique center of the smallest enclosing circle (SEC) of all robots, and then concurrently move there, finishing in one single round. Hence, we study the distributed version of the problem where each robot has a limited viewing range and can only observe other robots that are within unit distance of its position. This notion implies that the visibility graph of the robots is a unit disk graph (UDG). Clearly, the UDG of the robots must be connected initially, meaning that there is a path from any robot to any other robot just following the visibility neighborhoods. Additionally we assume that robots are anonymous, in the sense that they do not have unique IDs. Again, if robots have unique IDs, the problem becomes much simpler, as the robots just have to agree on meeting at the location of the robot with the minimum ID.

The most important question in the aforementioned model is whether the robots are able to meet at a single point and how long it takes to do so. The answer to the first question is known for 15 years. In their seminal paper, Ando, Suzuki, and Yamashita [2] presented an algorithm that gathers the robots. In each round, every robot simply moves to the center of the SEC of the robots in its viewing range, only constrained by the condition that robots must not lose visibility to their neighboring robots. As Ando et al. proved, this approach works, and the robots eventually meet.

More recently, Chazelle [6] showed that similar processes may have an exponential behavior. It is therefore an interesting task to examine runtime bounds of the original SEC algorithm by Ando et al. In this paper we show that the algorithm gathers all robots at a single point in a number of rounds polynomial in the number of nodes n, in particular $\mathcal{O}(n^2)$. Furthermore, we give a matching lower bound of $\Omega(n^2)$ and altogether present a tight analysis of the SEC algorithm, showing that the algorithm needs $\Theta(n^2)$ rounds to gather all robots.

2. RELATED WORK

The problem of gathering a set of robots has gained a lot of interest during the last 15 years. In early work, all robots had a global view of the positions of the other robots [24, 25]. Several articles have been published for the fully asynchronous and continuous setting, where the robots do not have a common notion of time, and hence may also observe each other while moving. A promising approach seems to let all robots move to the *Weber* point that – unlike the center of gravity or the center of the SEC – is invariant to movements of robots towards it. However, Bajaj [4] showed that the Weber point cannot be computed because it involves calculating roots of high-order polynomials. Cieliebak et al. [8] gave an algorithm that solves the gathering problem if the robots are able to detect whether there is more than one robot at a given point (multiplicity detection). Cohen et al. showed that moving to the center of gravity of the robots leads to convergence, even in highly asynchronous models [9, 10]. Furthermore, Izumi et al. [19] showed exponential lower bounds for the convergence of a certain class of randomized algorithms.

We are mainly interested in the local model with limited visibility, where the robots have to base their decisions only on the positions of the neighboring robots within a given range. This setting is more difficult, because a robot does not know the system as a whole, often not even the total number of robots. Furthermore, it is essential to always guarantee the connectivity of the neighborhood graph, given that it is connected in the beginning. Otherwise it cannot be ensured that the connectivity will ever be regained. This is especially an issue in a synchronous and discrete round model, which is common in the literature [2, 15, 24] and which we also consider in this paper. As the robots move at the same time (possibly based on different information), it is difficult to keep the connectivity.

The gathering problem in the local setting was already tackled some time ago by Ando et al. [2]. Similar to other local algorithms for the gathering problem, their robots move to the center of the smallest enclosing circle of their neighbors' locations. This target point definition guarantees that connectivity is maintained if no two robots are activated at the same time. But it can be easily seen that connectivity is not necessarily maintained in the synchronous setting. To overcome this problem, the authors restrict the distance that a robot moves towards its target point in a clever way, such that connectivity is guaranteed even under worst-case movement of the other robots performing the same algorithm. Furthermore, Ando et al. showed that their algorithm allows the robots to gather in a finite number of rounds. Beyond this result, no runtime bounds were given. A follow-up article [3] evaluated the quality of their algorithm in a more realistic environment, where sensor data is not perfectly accurate, and suggested that the algorithm is robust against measurement errors of the sensors.

The same algorithm, but in an asynchronous setting, was used by Meyer auf der Heide et al. [22]. Here, the robots only move one at a time, and so no connectivity maintenance is required. It is shown that the robots also gather in this setting, but again, no runtime bounds are given.

Flocchini et al. [17] showed that having a common orientation among the robots is sufficient to solve the gathering problem in finite time in the fully asynchronous model. The work by Degener et al. [13] is closest to our new contribution. It is shown that gathering can be achieved in expected $\mathcal{O}(n^3 \log n)$ rounds if the robots move sequentially: in each step only one robot (chosen uniformly at random) is activated. Moreover, when active, robots do not only move themselves, but they need the additional capability to assign new target points to neighbors, which may then move as well. This is a very powerful assumption, since it enables a robot to move several robots to the same position and let them act like one single robot from then on.

Apart from this result, there are no runtime bounds known for other algorithms for the local gathering problem so far.

Other researchers have analyzed how the algorithms can cope with failure or inaccuracies of sensor readings. Among others, Souissi et al. [23] and Izumi et al. [18] presented algorithms that are able to deal with erroneous readings from a compass. Agmon et al. [1] studied algorithms that tolerate the crash of a single robot, and still are able to achieve gathering of the remaining robots.

The more general problem of constructing geometrical formations with a set of autonomous robots has also attracted a lot of research. Current work shows how these robots can form lines between fixed stations [12, 15, 16, 20, 21] or circles [5, 11].

In this paper, we provide a lower bound of $\Omega(n^2)$ and, as our main result, a matching upper bound of $\mathcal{O}(n^2)$ for the number of rounds required to gather the robots using the local algorithm for the synchronous setting presented by Ando et al. [2]. The robots used here are considerably weaker than those discussed in the work of Degener et al. [13], as they they cannot instruct any robots to move and are not allowed to view any further than their communication range.

Note, that the needed capabilities are quite restrictive compared to related work from robot formation problems. Other capabilities that are considered are for instance compasses [18, 23] and other time models such as the semi-synchronous model, where arbitrary subgroups of robots move synchronously [14].

3. MODEL DEFINITION

Problem description and notation.

Our model is essentially the one defined by Ando et al. [2]. Given a set $\mathcal{R}$ of n robots $r_1, \ldots, r_n$ in the Euclidean plane, the goal is to gather all robots in one point. A robot is represented as a singular point in the plane, which means that robots cannot block each other's views or paths. We use a discrete, synchronous time model: In each round t, $t \in \mathbb{N}_0$, all robots act synchronously at the same time. We call the positions $p_1(t), \ldots, p_n(t)$ of the robots at the beginning of round t the *configuration* at time t. When the round t under consideration is clear from the context, we will sometimes identify a robot r_i with its position $p_i(t)$. We further call the configuration at time 0 the *start configuration*. When we say *time* t, we refer to the beginning of round t. The (Euclidean) distance between two robots r_i and r_j is indicated by $d(p_i(t), p_j(t))$ or also by $d(r_i, r_j)$. Two robots r_i and r_j can see each other, if $d(r_i, r_j) \leq 1$, where we call r_i and r_j *neighbors* and the distance 1 the *viewing range* of the robots. The set of all neighbors of a robot r_i – its *neighborhood* – at time t is denoted as $N_t(r_i)$ or just $N(r_i)$ if the time is clear from the context. The notion of limited visibility induces a unit disk graph, the *visibility graph* $UDG_t = (\mathcal{R}, E_t)$, where $(r_i, r_j) \in E_t$ iff r_i and r_j are mutually visible at time t, i.e. $\mathrm{dist}(r_i(t), r_j(t)) \leq 1$. We will furthermore use the convex hull of a set of robot positions to which we will also refer by the convex hull of these robots.

We measure the quality of the algorithm by counting the number of synchronous rounds until the robots have gathered in one point. During each round, the robots act according to the *Look-Compute-Move* (LCM) model: First all robots synchronously observe their environment and determine the positions of their neighbors relative to their own position (Look-operation). During the Compute-operation, they use the observed positions as input for the algorithm described in Section 4. The algorithm outputs the point to which the robots move during the following Move-operation.

The algorithm is based on the smallest enclosing circle (SEC) of a point set $\mathcal{P}$ (which are robot positions in our context). Its center is the point that minimizes the maximum distance to any point in $\mathcal{P}$.

Robot model.

Our robots have a limited viewing range, they are oblivious, which means that they do not have a memory, they do not communicate and they do not use a common coordinate system. Moreover, they cannot be distinguished from each other – they are anonymous. On the other hand, we abstract from technical issues. In particular, we assume the robots to be able to measure positions of neighbors relative to their own position accurately, they can compute geometric properties and they can occupy the same position as other robots.

4. THE ALGORITHM

Algorithm 1 ALGORITHM OF ROBOT r_i in round t

{*compute target point*}
$\mathcal{R}_i(t) :=$ {all robots visible from r_i including r_i itself}
$\mathcal{C}_i(t) :=$ smallest enclosing circle of $\mathcal{R}_i(t)$
$c_i(t) :=$ center of $\mathcal{C}_i(t)$
{*keep connectivity*}
$\forall r_j \in \mathcal{R}_i(t) : m_j :=$ the midpoint between $p_i(t)$ and $p_j(t)$
$\forall r_j \in \mathcal{R}_i(t) : \mathcal{D}_j :=$ the circle with radius $\frac{1}{2}$ around m_j
seg := the line segment $\overline{p_i(t), c_i(t)}$
$\mathcal{A} := \bigcap_{r_j \in \mathcal{R}} \mathcal{D}_j \cap \mathrm{seg}$
$x :=$ the point in A that minimizes $d(x, c_i(t))$
{*Note that* $\mathcal{A} \neq \emptyset$*, since* $p_i(t) \in \mathcal{A}$}
$p_i(t+1) := x$

The algorithm, which was introduced in [2], works as follows. First, r_i computes its *target point* $c_i(t)$, which is the center of the smallest enclosing circle around itself and its neighbors. Because the connectivity of the unit disk graph could break if all robots would move to their target point, a second phase is used to compute a point x on the line segment between $p_i(t)$ and $c_i(t)$ to which r_i finally moves. For each neighbor r_j, r_i computes the midpoint m_j between their positions and the *limit circle* D_j with center m_j and radius $1/2$. As long as both r_i and r_j do not leave this circle, they will be in distance 1 of each other and therefore neighbors at the beginning of the next round. Finally, x is the point on the line segment between $p_i(t)$ and $c_i(t)$ that maximizes the distance that r_i moves under the constraint that r_i does not leave the circle D_j for any neighbor r_j. Since all robots execute this algorithm, this procedure makes sure that two neighboring robots never lose their connection.

LEMMA 4.1 (ANDO ET AL. [2]). *If two robots are neighbors in* UDG_t *at time* t*, then they are still neighbors in* UDG_{t+1}*. In particular, if* UDG_0 *is connected, then* UDG_t *is connected for all* $t \geq 0$*.*

Because of the procedure to keep connectivity, it is possible that a robot does not move far in direction towards its target point. We say that a robot r_j *hinders* another robot r_i from reaching some point p on the line segment between $p_i(t)$ and $c_i(t)$, if r_i would leave D_j when moving to p. If in any round, two robots move to the exact same point, they will stay at a common point for the rest of the execution of the algorithm, because they see the same neighborhood and hence behave exactly the same. We call such robots to have *merged*.

In [2], the authors have already shown that this algorithm gathers the robots in one point within finite time, but so far no runtime bounds were known. We will now first show a lower bound $\Omega(n^2)$, and then our main result, namely the upper runtime bound of $\mathcal{O}(n^2)$ rounds.

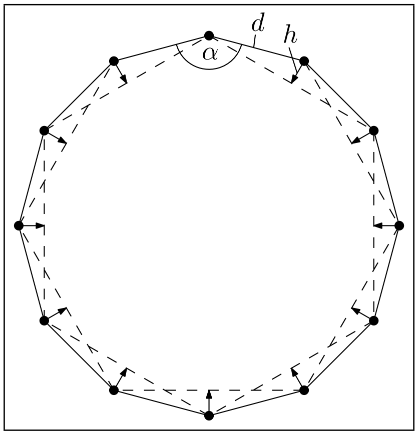

Figure 1: A robot configuration on the vertices of a regular convex polygon yields a worst-case running time of the algorithm.

5. THE LOWER BOUND

For a lower bound on the number of rounds until gathering when using the algorithm described in Section 4, consider a configuration with the robots positioned on the boundary of a circle, such that each robot has only two neighbors and the distance between two neighbors on the circle is the same for all robots. In this configuration, all robots have the same local view and so all robots do the same. The robots will therefore still be positioned on the boundary of a circle in the next round. We will use this observation to prove the following result.

THEOREM 5.1. *There is a start configuration such that the algorithm takes $\Omega(n^2)$ rounds to gather the robots in one point.*

PROOF. Let the robots be positioned on a circle with an initial distance of 1 between two neighboring robots (see Figure 1 for an illustration). This means that the initial circumference of the circle is $\approx n$, and its radius is $\approx \frac{n}{2\pi}$. We will show that it takes $\Omega(n^2)$ rounds until the circumference of the circle is reduced to $\frac{2}{3}n$.

If the circumference of the circle is greater than $\frac{2}{3}n$, each robot r has only two neighbors, which are in equal distance d, $\frac{1}{2} < d \leq 1$, from r. The center of the SEC of r's neighborhood is the midpoint between its neighbors. We can therefore compute the distance that r moves as the height h of the equilateral triangle formed by r and its two neighbors. To compute h, let α be the internal angle of the triangle at robot r. Due to the definition of the cosine, $h = \cos(\frac{\alpha}{2}) \cdot d$. In the interval between 0 and $\frac{\pi}{2}$, the cosine can be upper bounded by $\cos(x) \leq -x + \frac{\pi}{2}$. As $0 < \frac{\alpha}{2} < \frac{\pi}{2}$, we can apply this bound and thus $\cos(\frac{\alpha}{2}) \leq -\frac{\alpha}{2} + \frac{\pi}{2}$, resulting in $h \leq \left(-\frac{\alpha}{2} + \frac{\pi}{2}\right) \cdot d$. Moreover, since the robots form a regular polygon with n vertices and the sum of the internal angles of such a polygon is $\pi n - 2\pi$, we get that $\alpha = \pi - \frac{2\pi}{n}$ for all robots. Thus,

$$\begin{aligned} h &\leq \left(-\frac{\alpha}{2} + \frac{\pi}{2}\right) \cdot d \\ &\leq \left(-\left(\frac{\pi}{2} - \frac{\pi}{n}\right) + \frac{\pi}{2}\right) \cdot d \\ &= \frac{\pi}{n} \cdot d \leq \frac{\pi}{n} \end{aligned}$$

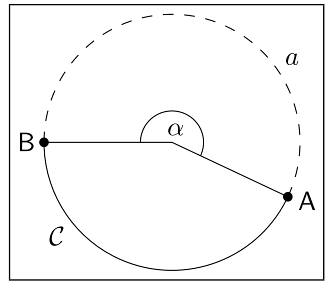

Figure 2: The central angle α of an arc a of the circle $\mathcal{C}$ is the angle subtended at the center of $\mathcal{C}$ by the two points A and B delimiting the arc.

and the robots move at most a distance of $\frac{\pi}{n}$ in each round. Therefore, it takes at least $\frac{1}{3\pi}n^2$ rounds until the radius is decreased by at least $\frac{1}{3}n$. As the circumference is 2π times the radius of a circle, decreasing the radius by $\frac{1}{3}n$ also decreases the circumference by $\frac{1}{3}n$. Thus, it takes at least $\frac{1}{3\pi}n^2$ rounds until the circumference is decreased to $\frac{2}{3}n$. $\square$

6. THE UPPER BOUND

In this section we will show that the robots gather in $\mathcal{O}(n^2)$ rounds. But before we start with the analysis, we state some well-known facts about smallest enclosing circles, on which our analysis will rely heavily.

PROPOSITION 6.1 (CHRYSTAL [7]). *Let $\mathcal{C}$ be the smallest enclosing circle (SEC) of a point set $\mathcal{S}$. Then either*

1. *there are two points $\mathsf{P}, \mathsf{Q} \in \mathcal{S}$ on the circumference of $\mathcal{C}$ such that the line segment $\overline{\mathsf{PQ}}$ is a diameter of $\mathcal{C}$, or*
2. *there are three points $\mathsf{P}, \mathsf{Q}, \mathsf{R} \in \mathcal{S}$ on the circumference of $\mathcal{C}$ such that the center c of $\mathcal{C}$ is inside $\triangle\mathsf{PQR}$, which means that $\triangle\mathsf{PQR}$ is acute-angled.*

Furthermore, the SEC of a set of points is unique.

From this proposition follows directly that the SEC of a point set P is always within the convex hull of P.

The following definition is illustrated in Figure 2.

DEFINITION 6.2. *Let $\mathcal{C}$ be the SEC of a set of points $\mathcal{S}$. An arc of $\mathcal{C}$ that contains no points is called a* point-free arc. *The* length *of this arc is defined as the central angle of the arc.*

Note that the central angle of an arc is greater than π if the arc extends over more than half the circumference of the circle.

PROPOSITION 6.3 (CHRYSTAL [7]). *Let $\mathcal{C}$ be the SEC of a set of $n \geq 2$ points. Then there is no point-free arc with length greater than π.*

With these basics, we can now define how we measure progress. We will use two progress measures.

- As a first progress measure, we will count the number of rounds in which robots merge. As we have n robots in the beginning, there can be at most $n - 1$ such rounds.

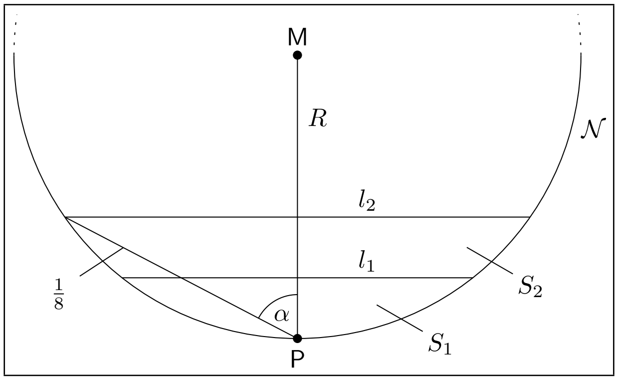

Figure 3: The segments S_1 and $S_1 \cup S_2$ of the global SEC are later used to measure the progress of the algorithm.

- Since the algorithm is deterministic and it was already proven in Ando et al.'s original paper [2] that the robots gather in finite time, we know that, for a given start configuration, the point where the robots gather is fixed. We will call this point the *gathering point* M. We define a circle $\mathcal{N}_t$ with center M and radius R_t for a round t, such that $\mathcal{N}_t$ contains all robots in round t and its radius is minimal. Due to the definition of the algorithm and because the center of the SEC of a point set is always within the convex hull of the point set, the robots never leave the convex hull of their neighbors as well as the global convex hull. R_t can therefore only decrease. We will use R_t as a second progress measure.

  As the robots gather at a point inside the convex hull of the robot positions in any round t, M is inside the convex hull of the robot positions of the start configuration. Moreover, since UDG_0 is connected, the diameter of the convex hull of the robots in round 0 can be at most $n-1$ and therefore also $R_0 \leq n-1$. The idea of the proof is to show that in a constant number of rounds in which no robots merge, R_t decreases by at least $\Omega(\frac{1}{n})$.

Using these two progress measures, with $R_0 \leq n-1$ and at most $n-1$ rounds in which robots merge, it follows directly that the robots gather in $\mathcal{O}(n^2)$ rounds.

From now on, we will consider an arbitrary but fixed round t_0. Let $\mathcal{N} := \mathcal{N}_{t_0}$ and $R := R_{t_0}$. For this round, we introduce some further notions (see Figure 3): first, fix an arbitrary point P on the boundary of $\mathcal{N}$ and draw a line between P and M. A line l_2 that is perpendicular to this line defines a circular segment of $\mathcal{N}$. The intersection points of l_2 and the circle $\mathcal{N}$ are in distance $\frac{1}{8}$ from P. Observe that the length of l_2 is bounded by $\frac{1}{4}$. We call S_1 the circular segment with half the height of the segment defined by l_2, such that a line l_1 that is parallel to l_2 is its chord. Moreover, we define S_2 to be the area of the segment defined by l_2 minus the area of S_1. The main idea of the analysis is to show that in round t_0 and t_0+1, either two robots merge or all robots leave S_1. We will conclude that this leads to the desired number of rounds.

The following analysis is divided into geometric prerequisites regarding S_1 and S_2 (Section 6.1) and the actual analysis of the algorithm (Section 6.2).

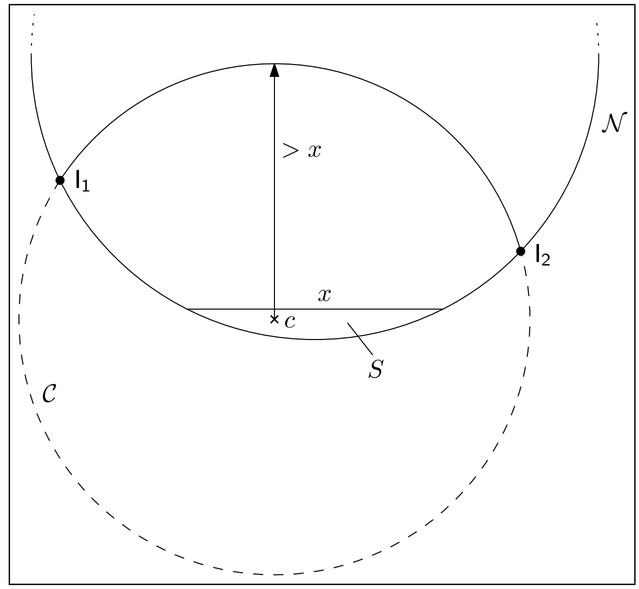

Figure 4: A circle with center in S and a radius exceeding the chord length of S intersects with $\mathcal{N}$ outside of S.

6.1 Geometric Prerequisites

In this section we want to give prerequisites regarding S_1 and S_2 and smallest enclosing circles with centers in these segments. These will be used later to make a statement about which robots can compute target points inside one of the segments.

LEMMA 6.4. *Let x be the length of a chord defining a circular segment S of $\mathcal{N}$. Then any circle $\mathcal{C}$ with its center c in S and radius $r > x$ has an arc outside of $\mathcal{N}$ with a central angle larger than π and thus cannot be the SEC of points only from $\mathcal{N}$.*

PROOF. See Figure 4 for an illustration of the setting described by the lemma. Since r is larger than the length of the maximum distance between two points in S, both intersection points I_1 and I_2 of the circle $\mathcal{N}$ with any circle with center in S and radius $r > x$ lie outside of S. Because the center c lies in S, it follows that the (longer) arc of $\mathcal{C}$ from I_1 to I_2 outside of $\mathcal{N}$ has a central angle larger than π (the dashed part of the circumference in Figure 4). □

Since the chord length of $S_1 \cup S_2$ is bounded by $\frac{1}{4}$, the following corollary is immediate.

COROLLARY 6.5. *The radius of a SEC of a point set $\mathcal{S} \subseteq \mathcal{N}$ with its center in $S_1 \cup S_2$ is at most $\frac{1}{4}$.*

In the following, we will show two geometrical lemmas for the position of the center of a SEC, if the configuration of the underlying points adheres to a few restrictions. The first lemma follows from Corollary 6.5 and will be used to show that if a robot can see a robot that is far away from $S_1 \cup S_2$, it cannot compute a target point inside this circular segment.

LEMMA 6.6. *Let $\mathcal{S} \subseteq \mathcal{N}$ be a set of points. Now let A be a point in $S_1 \cup S_2$ and $\mathsf{B} \in \mathcal{S}$ be a point in distance at least 1 from A. Then the center of the SEC of $\mathcal{S}$ cannot lie in the segment $S_1 \cup S_2$.*

Note that A does not need to be in $\mathcal{S}$.

PROOF. Assume that the SEC $\mathcal{C}$ has its center c inside $S_1 \cup S_2$. We know from Corollary 6.5 that $\mathcal{C}$ can have at most radius $\frac{1}{4}$. Since the maximum distance of two points in $S_1 \cup S_2$ is bounded by $\frac{1}{4}$, B must have a distance of at least $\frac{3}{4}$ from $S_1 \cup S_2$ in order to be in distance at least 1 from A. Hence, B cannot lie in $\mathcal{C}$. $\square$

The next lemma is similar to the last one in the sense that it makes a statement about configurations, for which robots cannot compute a target point in S_1. In particular, it will be used for robots that can only see one single robot in $S_1 \cup S_2$. These robots cannot compute a target point in S_1.

LEMMA 6.7. *The center of the SEC of a non-empty point set $\mathcal{S} \subseteq \mathcal{N} \setminus (S_1 \cup S_2)$ and a point $\mathsf{A} \in S_1 \cup S_2$ cannot lie in the segment S_1.*

PROOF. Assume that the SEC $\mathcal{C}$ has its center c inside S_1. We distinguish two cases as given by Proposition 1.

1. $\mathcal{C}$ is defined by two points P_1 and P_2. A must be one of these points, say P_2, otherwise c cannot lie in S_1. Since P_1 cannot lie in S_1 or S_2 by assumption and because the height of S_1 is equal to the height of S_2, the midpoint c of $\overline{\mathsf{AP}_1}$ cannot lie in S_1.

2. $\mathcal{C}$ is defined by three points P_1, P_1 and P_3. A must be one of these points, say P_3, otherwise c cannot lie in S_1. Since $\mathcal{C}$ is the circumcircle of $\triangle \mathsf{P}_1\mathsf{P}_2\mathsf{A}$, it lies on the intersection of the perpendicular bisectors of $\overline{\mathsf{AP}_1}$ and $\overline{\mathsf{AP}_2}$. The centers of these two segments lie outside S_1 and since the perpendicular bisectors intersect in the interior of $\triangle \mathsf{P}_1\mathsf{P}_2\mathsf{A}$ and this triangle is acute, their intersection point also cannot lie in S_1.

This completes the proof. $\square$

Finally, as the main idea of the analysis is to show that if no robots merge, S_1 is empty after two rounds, we will need the height of S_1 to compute the progress with respect to R_t within two rounds.

LEMMA 6.8. *The segment S_1 has a height h of at least $\frac{1}{128\pi \cdot R} \in \Omega\left(\frac{1}{n}\right)$.*

PROOF. We start by computing the angle α (see Figure 3 for a definition of α). The circumference of $\mathcal{N}$ is $2\pi R$. Thus, we can position at most $16\pi R$ points on the boundary of $\mathcal{N}$ that are in distance $\frac{1}{8}$ from the points closest to them and that form a regular convex polygon. The internal angle of each of the points of this polygon is equal to 2α. To compute such an internal angle, we use that the sum of the internal angles of a convex polygon is $(m-2) \cdot \pi$, where m is the number of vertices of the polygon. In our case, this is at most $(16\pi R - 2) \cdot \pi$. It follows that each angle is at most $\frac{(16\pi R-2)\cdot\pi}{16\pi R} = \pi - \frac{1}{8R}$, and thus $\alpha \leq \frac{\pi}{2} - \frac{1}{16R}$.

Now we can use α and the fact that $\cos(x) \geq -\frac{2}{\pi}x + 1$ in the interval $x \in [0, \frac{\pi}{2}]$ to compute the height h of S_1:

$$\begin{aligned} h = \frac{\cos\alpha}{16} &\geq \frac{\cos\left(\frac{\pi}{2} - \frac{1}{16R}\right)}{16} \\ &\geq \frac{1}{16} \cdot \left(-\frac{2}{\pi} \cdot \left(\frac{\pi}{2} - \frac{1}{16R}\right) + 1\right) \\ &= \frac{1}{128\pi R} \end{aligned}$$

Because $R \leq n$, we have shown $h \in \Omega(\frac{1}{n})$. $\square$

6.2 Gathering Algorithm Analysis

Now we can proceed to the actual analysis of the algorithm. We can use the lemmas from Section 6.1 to determine robots that cannot compute a target point in S_1 or $S_1 \cup S_2$. Nevertheless, according to the algorithm, robots do not always reach their target point; it is also possible that they are hindered by other robots. So knowing that a target point is outside S_1 or $S_1 \cup S_2$ does not necessarily mean that the robot actually leaves the respective segment. The following two lemmas show that robots always reach their target point, if it is in $S_1 \cup S_2$, and that they cannot be hindered from leaving S_1 and S_2.

LEMMA 6.9. *Robots that compute a target point in $S_1 \cup S_2$ cannot be hindered from reaching it by the limit circle of any other robot.*

PROOF. Let r_i be a robot that computes a target point c (which is the center of the SEC $\mathcal{C}$) inside $S_1 \cup S_2$. Then, according to Corollary 6.5, the radius of $\mathcal{C}$ cannot exceed $\frac{1}{4}$ and thus the distance between r_i and c is also upper bounded by $\frac{1}{4}$. Now assume that there is a robot r_e that hinders r_i from reaching c. Since r_e must be a neighbor of r_i, it must also be included in $\mathcal{C}$ and therefore, r_e can have at most distance $\frac{1}{2}$ from r_i. Now let m_e be the midpoint between r_i and r_e and therefore the center of the limit circle that hinders r_i from reaching c. m_e can be at most in distance $\frac{1}{4}$ from r_i. But that means that r_i can move freely in any direction a distance of $\frac{1}{2} - \frac{1}{4} = \frac{1}{4}$ and hence it can reach its target point without being hindered by r_e. $\square$

LEMMA 6.10. *Robots cannot be hindered from leaving $S_1 \cup S_2$ by the limit circle of any other robot.*

PROOF. Let r_i be a robot that computes a target point outside $S_1 \cup S_2$ in round t_0. Now assume for the sake of contradiction that there is one robot r_j that hinders r_i from leaving $S_1 \cup S_2$. This is only possible if r_j is a neighbor of r_i and thus r_j must be within distance 1 of r_i (see the circle $\mathcal{C}_1$ in Figure 5 with center r_i and radius 1: r_j must be in $\mathcal{C}_1$). Now let m be the point where r_i would leave $S_1 \cup S_2$ if moving to its target point. According to the algorithm it is only possible that r_i is hindered by r_j to leave $S_1 \cup S_2$, if m is not within distance $\frac{1}{2}$ from the midpoint m_j between r_i and r_j (line 6 – 10 of the algorithm). It follows that m_j cannot be inside the circle $\mathcal{C}_2$ (Figure 5) with center m and radius $\frac{1}{2}$. Based on $\mathcal{C}_2$ we can define a circle $\mathcal{C}_3$ which may not contain r_j, if m_j is not in $\mathcal{C}_2$: $\mathcal{C}_3$'s center is p_i', which is p_i reflected with respect to the point m, and its radius is 1 (see Figure 5). Summing up, r_j must be inside of $\mathcal{C}_1$, but outside of $\mathcal{C}_3$. Moreover, the smallest enclosing circle computed by the algorithm has at most radius 1, and so r_i's target point is at most in distance 1 of r_j. It follows that r_i's target point must be on the line between m and p_i', because

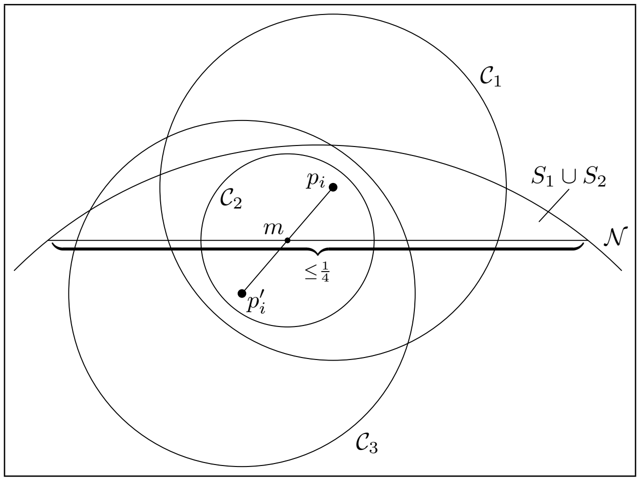

Figure 5: Illustration of the proof of Lemma 6.10. The circles indicate where r_j can be positioned: $\mathcal{C}_1$ is a circle with center p_i and radius 1 and must contain r_j. $\mathcal{C}_2$ has center m and radius $\frac{1}{2}$, and $\mathcal{C}_3$'s center is p'_i with radius 1. r_j must not be in $\mathcal{C}_3$.

each point on the straight line through p_i and m beyond p'_i is in distance more than 1 from any point that is in $\mathcal{C}_1$, but not in $\mathcal{C}_3$.

Case 1: r_j is in $S_1 \cup S_2$. Then, because the chord length of $S_1 \cup S_2$ is at most $\frac{1}{4}$, the distance between r_i and r_j is also at most $\frac{1}{4}$. But that means that r_i is at most in distance $\frac{1}{8}$ from the midpoint between r_i and r_j and thus it can move at least distance $\frac{1}{2} - \frac{1}{8} = \frac{3}{8} > \frac{1}{4}$ freely in any direction without being hindered by r_j. But after r_i has moved a distance of $\frac{1}{4}$, it has left $S_1 \cup S_2$ leading to a contradiction.

Case 2: r_j is not in $S_1 \cup S_2$. Since a SEC is defined by two or three points with at least one point on each half of the boundary of the SEC (Proposition 6.3), there must be a robot r_k that is in $S_1 \cup S_2$ and on the boundary of the SEC defining r_i's target point. It follows that r_k can be at most in distance $\frac{1}{4}$ from m. As p_i is also at most in distance $\frac{1}{4}$ from m, so is p'_i and also p_i's target point, which is between m and p'_i (see above). Thus, r_k is at most in distance $\frac{1}{2}$ from r_i's target point. Since r_k is on the boundary of the SEC that defines r_i's target point, it follows that the SEC can have at most a radius of $\frac{1}{2}$. Now, since r_j is outside of $\mathcal{C}_3$ and because the distance between m and p'_i is at most $\frac{1}{4}$ (see above), r_j must be in distance greater than $\frac{1}{2}$ from r_i's target point. Thus, r_j cannot be in the SEC that defines r_i's target point, which is a contradiction to r_i and r_j being neighbors. It follows that r_j cannot hinder r_i from leaving $S_1 \cup S_2$. □

With all these prerequisites, we can now show that if no robots merge, S_1 is empty after two rounds. We first analyze the behavior of some robots in round t_0 in Lemma 6.11, before we plug things together in Lemma 6.12.

LEMMA 6.11. *Let $\mathcal{S}$ be a set of robots in round t_0 that are all positioned in or compute a target point in $S_1 \cup S_2$ and that all have a pairwise different neighborhood. Then at most one of those robots is in $S_1 \cup S_2$ at the beginning of the next round.*

PROOF. Since all robots from $\mathcal{S}$ have different neighbors, there exists a robot $r_i \in \mathcal{S}$ for which no robot from $\mathcal{S}$ has a set of neighbors that is a subset of the neighbors of r_i. Thus, all robots $r_j \in \mathcal{S} \setminus \{r_i\}$ have a neighbor that is not visible from r_i and therefore in distance more than 1 from r_i. If r_i is positioned in $S_1 \cup S_2$, all robots $r_j \in \mathcal{S} \setminus \{r_i\}$ see a point B in $\mathcal{N}$ (namely the position of the neighbor that r_i cannot see) that is in distance 1 from a point A in $S_1 \cup S_2$ (namely the position of r_i). Lemma 6.6 therefore guarantees that all neighbors of r_i compute a target point outside of $S_1 \cup S_2$. According to Lemma 6.10, no robot is hindered from leaving $S_1 \cup S_2$. Thus, only r_i can stay in $S_1 \cup S_2$.

If r_i is positioned outside $S_1 \cup S_2$, it has its target point in $S_1 \cup S_2$ according to the definition of $\mathcal{S}$. Corollary 6.5 now gives that the radius of r_i's SEC cannot exceed $\frac{1}{4}$ and thus r_i is in distance at most $\frac{1}{4}$ from $S_1 \cup S_2$. Using that the distance between two points in $S_1 \cup S_2$ is at most $\frac{1}{4}$, it follows that all points within $S_1 \cup S_2$ are in distance at most $\frac{1}{2}$ from r_i. Now consider a robot $r_j \in \mathcal{S} \setminus \{r_i\}$ and a neighbor r_k of r_j that is in distance more than 1 from r_i. This robot r_k must then be in distance more than $\frac{1}{2}$ from $S_1 \cup S_2$. Since r_k is r_j's neighbor, we know from Corollary 6.5, that the center of r_j's SEC – its target point – cannot be in $S_1 \cup S_2$ and according to Lemma 6.10 r_j is not hindered from leaving $S_1 \cup S_2$. Since this holds for all robots $r_j \in \mathcal{S} \setminus \{r_i\}$, r_i is the only robot that can be in $S_1 \cup S_2$ in round $t+1$. □

LEMMA 6.12. *If $R_t \geq \frac{1}{2}$, either there are robots that merge in round t or after two rounds, the segment S_1 does not contain any robots.*

PROOF. We consider all robots that are positioned in $S_1 \cup S_2$ or compute a target point in $S_1 \cup S_2$ in round t. We divide this set of robots into two subsets and analyze them separately.

- First, we consider all robots that have a neighbor with the same neighborhood. Thus, for all these robots there is another robot that computes the same target point. Then there are two possibilities: Either one of these target points is in $S_1 \cup S_2$. According to Lemma 6.9, the robots with this target point are not hindered from reaching it and therefore they merge. If all target points are outside $S_1 \cup S_2$, Lemma 6.10 guarantees that all these robots leave $S_1 \cup S_2$.

- Now consider the robots that have a pairwise different neighborhood. According to Lemma 6.11, at most one of those robots stays in $S_1 \cup S_2$ during this round.

Thus, if r_i is positioned outside S_1 at the end of round t, we are done. Otherwise, since apart from r_i no robot is still in $S_1 \cup S_2$, we know from Lemma 6.7, that neither r_i nor a neighbor of r_i can compute a target point in S_1 in round $t+1$. Thus, r_i leaves S_1 in round $t+1$ (Lemma 6.10) and none of its neighbors enters S_1. All other robots that are not neighbors of r_i do not see a robot in S_1 and thus they cannot enter S_1. □

Lemma 6.12 will be used to show that if no robots merge, R_t decreases by $\Omega\left(\frac{1}{n}\right)$ every two rounds. According to the following Lemma, this procedure stops as soon as $R_t < \frac{1}{2}$.

LEMMA 6.13 (ANDO ET AL. [2]). *If $R_t < \frac{1}{2}$, the robots have gathered at one point in round $t+1$.*

This lemma holds because if $R_t < \frac{1}{2}$, all robots can see each other and thus all robots compute the same target point. It is shown Ando et al.'s original work [2] that the robots do not hinder each other from reaching this point.

Putting everything together, we are now able to prove the final result.

THEOREM 6.14. *The robots gather within $\mathcal{O}(n^2)$ rounds.*

PROOF. Fix an arbitrary round $t_0 \geq 0$. Since Lemma 6.12 holds for any point on the boundary of N_{t_0}, after two rounds either two robots have merged or all robots must be in distance greater than the height of S_1 from the boundary of N_{t_0}. According to Lemma 6.8, the height of S_1 is at least $\frac{1}{128 \cdot R_t}$ and thus if the robots do not merge, the radius decreases by at least $\frac{1}{128 \cdot R_t}$, giving that $R_{t+2} \leq R_t - \frac{1}{128 \cdot R_t} \leq R_t - \frac{1}{128 \cdot R_0}$. It follows that after $2 \cdot 128 \cdot (R_0)^2 = 256 \cdot (R_0)^2$ rounds without merging robots, the radius must be less than $\frac{1}{2}$. Now it takes one round to gather the robots (Lemma 6.13). Moreover, since UDG_0 is connected, $R_0 \leq n$. There are at most $n - 1$ rounds in which robots merge. The total number of rounds is therefore at most $256 \cdot n^2 + n$. $\square$

7. CONCLUSION AND OUTLOOK

In this paper we have shown that mobile robots can gather at a single point in $\mathcal{O}(n^2)$ rounds, when they execute the classic synchronous algorithm by Ando et al. [2]. Furthermore we showed that this bound is asymptotically tight for this algorithm. This raises the question whether there are more efficient algorithms.

On the other hand there are no nontrivial lower bounds known for classes of local algorithms for gathering or other formation problems. One would need a clean definition of asynchronous or synchronous local gathering strategies. A crucial property restricting such strategies is that connectivity has to be maintained. Just looking at the start configuration of the lower bound instance from Section 5, for example, and only demanding connectivity for this specific start configuration is not sufficient: consider the synchronous algorithm in which each point moves in the direction of the target point of our algorithm, but goes beyond this point until the distance to its neighbors is 1. This algorithm maintains connectivity for our specific start configuration, but needs only a linear number of rounds, if the start configuration positions neighboring robots in distance $\frac{2}{3}$ on the cycle. Similar results can be shown for asynchronous strategies with specific activation policies. Such examples demonstrate that the connectivity constraint has to be reflected much more severely in lower-bound models for local gathering strategies.

References

[1] Noa Agmon and David Peleg. Fault-tolerant gathering algorithms for autonomous mobile robots. In *Proceedings of the Fifteenth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2004*, pages 1070–1078, January 2004.

[2] Hideki Ando, Yoshinobu Suzuki, and Masafumi Yamashita. Formation and agreement problems for synchronous mobile robots with limited visibility. In *Proceedings of the 1995 IEEE International Symposium on Intelligent Control, ISIC 1995*, pages 453–460, August 1995.

[3] Hideki Ando, Yoshinobu Oasa, Ichiro Suzuki, and Masafumi Yamashita. Distributed memoryless point convergence algorithm for mobile robots with limited visibility. *IEEE Transactions on Robotics and Automation*, 15(5):818–828, 1999.

[4] Chanderjit Bajaj. The algebraic degree of geometric optimization problems. *Discrete & Computational Geometry*, 3:177–191, 1988.

[5] I. Chatzigiannakis, M. Markou, and S. Nikoletseas. Distributed circle formation for anonymous oblivious robots. In *Proceedings of the Third International Workshop on Experimental and Efficient Algorithms, WEA 2004*, pages 159–174, May 2004.

[6] Bernard Chazelle. Natural algorithms. In *Proceedings of the Twentieth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2009*, pages 422–431, January 2009.

[7] George Chrystal. On the problem to construct the minimum circle enclosing n given points in a plane. In *Proceedings of the Edinburgh Mathematical Society, Third Meeting*, pages 30–35, 1885.

[8] M. Cieliebak, P. Flocchini, G. Prencipe, and N. Santoro. Solving the robots gathering problem. In *Proceedings of the Thirtieth International Colloquium on Automata, Languages and Programming, ICALP 2003*, pages 1181–1196, June 2003.

[9] Reuven Cohen and David Peleg. Robot convergence via center-of-gravity algorithms. In *Proceedings of the 11th International Colloquium on Structural Information and Communication Complexity, SIROCCO 2004*, pages 79–88, June 2004.

[10] Reuven Cohen and David Peleg. Convergence properties of the gravitational algorithm in asynchronous robot systems. *SIAM Journal on Computing*, 34(6): 1516–1528, 2005.

[11] X. Défago and A. Konagaya. Circle formation for oblivious anonymous mobile robots with no common sense of orientation. In *Proceedings of the 2nd ACM International Workshop on Principles of Mobile Computing, POMC 2002*, pages 97–104, October 2002.

[12] Bastian Degener, Barbara Kempkes, Peter Kling, and Friedhelm Meyer auf der Heide. A continuous, local strategy for constructing a short chain of mobile robots. In *Proceedings of the 17th International Colloquium on Structural Information and Communication Complexity, SIROCCO 2010*, pages 168–182, June 2010.

[13] Bastian Degener, Barbara Kempkes, and Friedhelm Meyer auf der Heide. A local $O(n^2)$ gathering algorithm. In *Proceedings of the 22nd Annual ACM Symposium on Parallelism in Algorithms and Architectures, SPAA 2010*, pages 217–223, June 2010.

[14] Yoann Dieudonné and Franck Petit. Self-stabilizing deterministic gathering. In *Proceedings of the 5th International Workshop on Algorithmic Aspects of Wireless Sensor Networks, ALGOSENSORS 2009*, pages 230–241, July 2009.

[15] Miroslaw Dynia, Jarosław Kutyłowski, Paweł Lorek, and Friedhelm Meyer auf der Heide. Maintaining communication between an explorer and a base station. In *Proceedings of the 1st IFIP International Conference on Biologically Inspired Collaborative Computing, BICC 2006*, pages 137–146, August 2006.

[16] Miroslaw Dynia, Jarosław Kutyłowski, Friedhelm Meyer auf der Heide, and Jonas Schrieb. Local strategies for maintaining a chain of relay stations between an explorer and a base station. In *Proceedings of the nineteenth annual ACM symposium on Parallel algorithms and architectures, SPAA 2007*, pages 260–269, January 2007.

[17] Paola Flocchini, Giuseppe Prencipe, Nicola Santoro, and Peter Widmayer. Gathering of asynchronous robots with limited visibility. *Theoretical Computer Science*, 337(1–3):147–168, 2005.

[18] Taisuke Izumi, Yoshiaki Katayama, Nobuhiro Inuzuka, and Koichi Wada. Gathering autonomous mobile robots with dynamic compasses: An optimal result. In *Proceedings of the 21st International Symposium on Distributed Computing, DISC 2007*, pages 298–312, September 2007.

[19] Taisuke Izumi, Tomoko Izumi, Sayaka Kamei, and Fukuhito Ooshita. Randomized gathering of mobile robots with local-multiplicity detection. In *Proceedings of the 11th International Symposium on Stabilization, Safety, and Security of Distributed Systems, SSS 2009*, pages 384–398, November 2009.

[20] Peter Kling and Friedhelm Meyer auf der Heide. Convergence of local communication chain strategies via linear transformations. this conference.

[21] Jaroslaw Kutylowski and Friedhelm Meyer auf der Heide. Optimal strategies for maintaining a chain of relays between an explorer and a base camp. *Theoretical Computer Science*, 410(36):3391–3405, 2009.

[22] Friedhelm Meyer auf der Heide and Barbara Schneider. Local strategies for connecting stations by small robotic networks. In *Proceedings of the 2nd IFIP International Conference on Biologically Inspired Collaborative Computing, BICC 2008*, pages 95–104, September 2008.

[23] Samia Souissi, Xavier Défago, and Masafumi Yamashita. Gathering asynchronous mobile robots with inaccurate compasses. In *Proceedings of the 25th Annual ACM Symposium on Principles of Distributed Systems, PODC 2006*, pages 333–349, July 2006.

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

[25] Ichiro Suzuki and Masafumi Yamashita. Formation and agreement problems for anonymous mobile robots. In *Proceedings of the 31st Annual Allerton Conference on Communication, Control, and Computing*, pages 93–102, September 1993.