Eventually Consistent Compasses for Robust Gathering of Asynchronous Mobile Robots with Limited Visibility

Samia Souissi[1], Xavier Défago[1], and Masafumi Yamashita[2]

[1]*School of Information Science, Japan Advanced Institute of Science and Technology (JAIST)*
[2]*Dept. of Computer Science and Communication Engineering, Kyushu University, Fukuoka, Japan*

July 20, 2005

IS-RR-2005-010

Japan Advanced Institute of Science and Technology (JAIST)

School of Information Science
1-1 Asahidai, Nomi, Ishikawa 923-1292, Japan

http://www.jaist.ac.jp/

ISSN 0918-7553

Eventually Consistent Compasses for Robust Gathering of Asynchronous Mobile Robots with Limited Visibility

Samia Souissi* Xavier Défago* Masafumi Yamashita ‡

*School of Information Science
Japan Advanced Institute of Science and Technology (JAIST)
1-1 Asahidai, Nomi, Ishikawa 923-1292, Japan
Tel.: +81-761-51 1254 - Fax.: +81-761-51 1149
‡ Department of Computer Science and Communication Engineering,
Kyushu University, Fukuoka, Japan
E-mail: mak@csce.kyushu-u.ac.jp
E-mail: {ssouissi,defago}@jaist.ac.jp

Abstract

Reaching agreement among a set of mobile robots is one of the most fundamental issues in distributed robotic systems. This problem is often illustrated by the gathering problem, where the robots must self-organize to eventually meet at some arbitrary location. That problem has the advantage that, while being very simple to express, it retains the inherent difficulty of agreement, namely the problem of breaking symmetry. In their fully asynchronous model with oblivious robots and limited visibility, Flocchini et al. [7] show that gathering is solvable, as long as the robots share the knowledge of some direction, as provided by a compass. It turns out that, in robotic systems, compasses are devices that are often subject to instabilities. In this paper, we thus define a model with unreliable compasses and, focusing on the gathering problem, show that the algorithm of Flocchini et al. is unable to tolerate unstable compasses. We then give a gathering algorithm that solves the problem in a system where compasses are unstable for some arbitrary long periods, provided that they stabilize eventually.

Keywords: Mobile computing, Distributed algorithms, Robots, Unreliable compasses, Oblivious computations, Limited Visibility, Self-stabilization.

1 Introduction

The problem of reaching agreement among robots has attracted considerable attention within the last few years. However, most of the algorithmic results we are aware of do not consider cases when sensors are unreliable. Particularly, most of the work on asynchronous models for cooperative mobile robots relies on the assumption that compasses provide perfect information. However, these components are frequently prone to failures. For instance, we can imagine a magnetic field created near the robots, which would make their compasses point in erroneous directions. As a result, if the algorithms are not designed to tolerate such transient failures, the robots will not accomplish their given task. In this paper, we study the solvability of the gathering problem in one class of unreliable compasses. In particular, we look at this problem in the face of instability of the compasses and we provide a new deterministic algorithm for solving the gathering problem in an asynchronous system, where robots are oblivious (i.e., they don't remember past actions and observations), they have limited visibility and their compasses are unstable for a certain amount of time unknown to the robots. Our algorithm is guaranteed to recover from any arbitrary configuration when the compasses of the robots eventually stabilize. One can argue that our algorithm is intrinsically self-stabilizing [12][1] and offers protection against any number of transient failures.

Related work. Despite its apparent simplicity, the problem of gathering robots at a single point has been studied extensively in the literature, in different models and under several assumptions. In fact, several factors render this problem difficult to solve [3, 4, 7, 10, 12]. In particular, in all these studies, the problem has been solved only by making some additional assumptions on the capabilities of the robots. In particular, in the asynchronous model, Flocchini et al. [7] proposed a deterministic algorithm for the gathering problem in the limited visibility setting. However, the proposed algorithm requires that the robots share a compass that provides perfect information. In contrast, we study the solvability of the gathering problem in the face of compass instability. The work of Flocchini et al. [7] is the closest to our work, and our algorithm is developed on their model. However, their algorithm fails to solve the gathering problem if we assume instable compasses. Subsequently, we provide a self-stabilizing algorithm that tolerates any number of transient failure in the compasses. The gathering problem has been also studied in the presence of faulty robots by Agmon and Peleg [1] in synchronous and asynchronous settings. In particular, they proposed an algorithm that tolerates one crash-faulty robot in a system of three or more robots. They also showed that in an asynchronous environment it is impossible to perform a successful gathering in a 3-robot system with one Byzantine failure. Other studies of the gathering problem include the work of Suzuki and Yamashita [12]. In their model, refereed as a semi-synchronous model,[2] they proposed an algorithm to solve the gathering problem deterministically in the case where robots have unlimited visibility. In the same model, Ando et al. [2] propose an algorithm to address the gathering problem in systems wherein robots have limited visibility. Their algorithm converges toward a solution to the problem, but it does not solve it deterministically. In

[1] Self-stabilization is the property of a system which, starting in an arbitrary state, always converges toward a desired behavior [6, 11]

[2] Currently, three main models are considered in the literature, depending on the degree of synchrony between robots' activation. Agmon and Peleg [1] classified them as *synchronous*, *semi-synchronous* [12] and *asynchronous* [9]. The model of Suzuki and Yamashita [12] assumes that activations (observe, compute, move) occur atomically, resulting in a form of implicit synchronization. The model is called *semi-synchronous* model for this reason.

the CORDA model, referred as an asynchronous model[3] [9], Cielibak et al. [4] proposed a deterministic algorithm that gathers the robots at a point in systems in which robots have unlimited visibility. Among other things, one feature the robots must have in order to solve this problem is the ability to detect a multiplicity of robots at one point. Other studies of the gathering problem has been devoted to design convergence solutions to the problem [5].

Contribution. The main contribution of this paper is to consider an important agreement problem (gathering) in face of unreliable compasses. In particular, we study the solvability of the gathering problem with respect to one class of unreliable compasses, that is, eventually consistent compasses. We first show that the gathering algorithm proposed by Flocchini et al. [7] can not cope with instabilities in the compasses. Second, we propose a self-stabilizing distributed algorithm for the gathering problem in a system in which the robots have limited visibility, relying on compasses that are eventually consistent. The proposed solution guarantees that the robots gather at a point in finite time, given that their compasses provide correct output after some unknown period of instability, during which our algorithm can tolerate any number of transient failures of the compasses.

Structure. The remainder of this paper is organized as follows. In Section 2, we introduce the system model and some definitions used in this paper. In Section 3, we define the concept of compass. In Section 4, we show that the algorithm of Flocchini et al. [7] does not solve the gathering problem assuming instable compasses. In Section 5, we describe our gathering algorithm based on eventually consistent compasses and in Section 6, we prove its correctness. Finally, in Section 7, we conclude the paper.

2 System Model and Definitions

2.1 CORDA model

We consider an asynchronous system of autonomous mobile robots roaming on the plane, in particular, the CORDA model [8, 9] defined by Prencipe. In this model, each robot is modelled as a component with computational capabilities; it is able to sense its surroundings, perform computations on the sensed data, and move toward the computed destination. This constitutes its computation cycle: inactive, sensing, computing, and moving. More precisely, the model is described as follows. Each robot is modelled as a point on the Euclidean plane, and it has its own local view of the world. This view includes an origin, a unit distance, and the directions and orientations of two x and y coordinate axes. Robots do not have agreement, neither on the origin nor on a unit distance. However they agree on the orientations and directions of the x and y axes, which means that the robots are *oriented*. For instance, the robots share a *compass*. The robots are *anonymous*, in the sense that they can not be distinguished by their appearance. In addition, there is no explicit direct means of communication among them. Hence, the only way for robots to acquire information is by observing each other's positions. The robots execute the same deterministic algorithm, which takes as input the positions of the visible robots, and returns a destination point toward which the robot moves. Moreover, the robots are totally asynchronous, and they do rely neither on any centralized directives nor on any common notion of time. We further assume that the robots have *limited visibility*, and hence

[3]The CORDA model [9], assumes that activations (wait-observe, compute, move) occur in a fully asynchronous manner. The model is called the *asynchronous* model for this reason.

each robot can see only the robots which are within its visibility radius V[4] , and the robots are *oblivious* or memoryless, which implies that they are unable to remember past actions and observations, and thus their computations can not be based on previous observations. The computation cycle of a robot is as follows: a robot is initially in a *waiting* state (*Wait*). Asynchronously and independently from the other robots, it *observes* the environment in its visibility radius (*Look*) by taking a snapshot of the positions of the robots within its visibility radius with respect to its local coordinate system[5]. Then, it *computes* a destination point based on the observed positions of the robots (*Compute*)[6]. Finally, the robot moves toward its computed destination (*Move*). If the destination is the current location, the robot is said to perform a *null movement*; otherwise, it is said to execute a *real movement*. The speed of the robot is not necessarily uniform. In addition, the move can end anywhere before the destination point. Finally, after moving, the robot goes back to the waiting state. The sequence *Wait-Look-Compute-Move* is called the *cycle* of a robot. The robots can be partitioned into sets depending on their state at a given time:

- $W(t)$ and $L(t)$ are the sets of all robots that are respectively in state *Wait* and *Look* at time t.
- $C(t)$ is the set of all the robots that at time t are in state *Compute*; the subset $C_{\varnothing}(t)$ contains the robots whose computation results in executing a *null movement*.
- $M(t)$ is the set of all the robots that at time t are executing a movement; the subset $M_{\varnothing}(t)$ contains the robots executing a *null movement* (they stay still).

In the model, there are two limiting assumptions related to the cycle of a robot.

Assumption 1 *It is assumed that the distance travelled by a robot r in a move is not infinite. Furthermore, it is not infinitesimally small: there exists a constant $\delta_r > 0$, such that, if the target point is closer than δ_r , r will reach it; otherwise, r will move toward it by at least δ_r.*

Note that without this assumption, it would be impossible for any algorithm to terminate in a finite time.

Assumption 2 *The amount of time required by a robot r to complete a cycle (wait-look-compute-move) is not infinite. Furthermore, it is not infinitesimally small; there exists a constant $\epsilon_r > 0$, such that the cycle will require at least ϵ_r time.*

2.2 Definitions

We now introduce further definitions and lemmas that come from Flocchini et al. [7].

Definition 1 (Distance graph) *Let $G = (N, E)$ indicate the distance graph of the robots, where the nodes represent the robots, and the edges represent the distance between the pairs of robots. Informally, the graph G is connected if for each pair (r, r') of robots in the set of robots N, the segment $\overline{rr'}$ belongs to the set of edges E, and the distance between r and r' is less or equal to V. Formally, G is connected if $\forall (r, r') \in N, (\overline{rr'} \in E) \wedge (\overline{rr'}) \leq V$.*

[4]We assume that all the robots have the same visibility radius.

[5]Note that a robot is unable to detect multiplicity, i.e., whether there is one or more robots on any of the observed points, including the position where the observing robot is.

[6]The destination point could be the current position of the robot.

Lemma 1 *If the distance graph is disconnected, the gathering problem is unsolvable.*

Definition 2 (Mutual visibility) *Informally, two robots r and r' are* mutually visible *at time t, if they include each other in their computations. Formally, two robots r and r' are* mutually visible *at time t if and only if the following both conditions hold:*

1. $0 < \overline{r(t)r'(t)} \leq V$
2. $r, r' \in L(t) \bigcup C_{\varnothing}(t) \bigcup M_{\varnothing}(t) \bigcup W(t)$

Since initially all the robots are in a waiting state, then all the pairs of robots that are initially within distance V from each other are initially mutually visible. Note that the definition of mutual visibility does not include the robots that are on the same point.

2.3 Notations

We denote by $\mathcal{U}$ the *universe*, including all the robots in the system. Given some robot r, $r(t)$ is the position of r at time t. $Look(r, t)$ denotes the *look* operation performed by robot r at time t. As a result, the circle $\mathcal{C}_r(t)$, centered at r, and with radius V denote the circle of visibility of r at time t. When no ambiguity arises, we shall omit the temporal indication.

Let A and B be two points; with $\overline{AB}$, we will indicate the segment starting at A and terminating at B. When no ambiguity arises, we will also use the notation $\overline{AB}$ to denote the length of such segment. By construction, we denote by $C(r, R)(t)$, the circle centered at r, and with radius R at time t. Also, we denote by $C(\overline{rr'})$, the circle with diameter $\overline{rr'}$ passing from robot r and r', provided that $r \neq r'$. Given a region X, we denote by $|X|$, the number of robots in that region at time t. Finally, given three distinct point A, B, and C, we denote by $\triangle(A, B, C)$, the triangle having them as corners.

We now introduce some important lemmas that will be used later in the paper.

2.4 Geometric properties

Lemma 2 *Every internal chord of a circle has a length less than or equal to the diameter. That is, the distance between any two points that belong to a circle is less or equal to the diameter.*

Lemma 3 *Any point that belongs to an internal chord of a circle is located inside the circle.*

Lemma 4 *Every internal chord of a triangle has length less than or equal to the longest side of the triangle.*

Lemma 5 *Given three points A, B and D with distinct positions and non collinear, and two circles $C(\overline{AB})$ and $C(\overline{AD})$ that intersect in two points A and H. Then, B, H and D are collinear.*

PROOF. H belongs to the circle $C(\overline{AB})$ and $C(\overline{AD})$ by definition (refer to Figure 1). Consequently, the angle $\angle AHB$ is angular on H since $\overline{AB}$ is the diameter of the circle $C(\overline{AB})$, and $H \in C(\overline{AD})$. Similarly, the angle $\angle AHD$ is angular on H. Hence, the measure of the angle $\angle BHD$ is equal to $180°$, and B, H and D are collinear, and the lemma holds. $\square_{\text{Lemma 5}}$

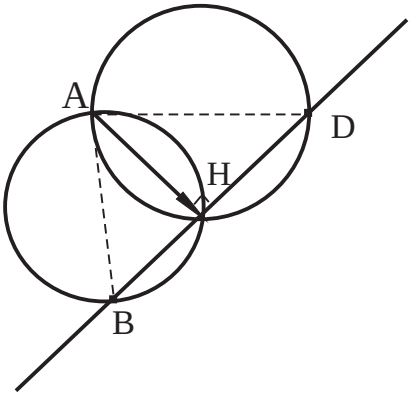


Figure 1. The circles $C(\overline{AB})$ and $C(\overline{AD})$ intersect on A and H: H is collinear with B and D.

3 Compasses

3.1 Definition

Definition 3 (Compass) *A compass is a function of time and robots. The function outputs a vector of direction (for instance, north).* $compass_r(t)$ *denotes the orientation of the compass of robot* r *at time* t*.*

By looking at the property of the compass, we derive the following two classes:

3.2 Perfect compass

With a *perfect compass*, the robots always agree on the same direction, i.e, the robots agree on the directions and orientations of both x and y axes. In other words, the directions of the compasses of the robots are the same at any time t. Formally, the robots on the universe $\mathcal{U}$ share a *perfect compass* if and only the *agreement* and *invariance* properties are satisfied:

Definition 4 (Perfect compass) *A perfect compass is defined by the following two properties:*

1. *Agreement:* $\forall r, r' \in \mathcal{U}, \forall t, compass_r(t) = compass_{r'}(t)$
2. *Invariance:* $\forall r \in \mathcal{U}, \forall t, t', compass_r(t) = compass_r(t')$

3.3 Eventual compass

With an *eventual compass*, there exists a time after which all the robots agree on the same compass direction. The agreement holds after some time GST (Global Stabilization Time) unknown to the robots. It is only guaranteed that the agreement will hold, but the time for which the agreement holds is unknown. More precisely, an eventual compass has the following properties: (1) The direction of a robot's compass can change with time. (2) At a given time, the compasses of any two robots may disagree. (3) In any execution, there exists some time GST after which the compasses of all the robots agree for a sufficiently long period. Yet, the robots does not know

when the time GST will occur. Formally, the robots on $\mathcal{U}$ share an *eventual compass* if and only if the *eventual agreement* and *invariance* properties hold:

Definition 5 (Eventual compass) *An* eventual compass *has the following two properties:*

1. *Eventual agreement:* $\exists GST, \forall r, r' \in \mathcal{U}, \forall t \geq GST, compass_r(t) = compass_{r'}(t)$
2. *Invariance:* $\forall r \in \mathcal{U}, \forall t, t' \geq GST, compass_r(t) = compass_r(t')$

We further introduce the following notations. We denote by $Left_r(t)$, $Right_r(t)$, $Top_r(t)$ and $Bottom_r(t)$ respectively, the left, right, top and bottom side according to the orientation of the compass of some robot r at time t. Note that, the robot r is not included in any of the sides. When no ambiguity arises, we shall omit the temporal indication.

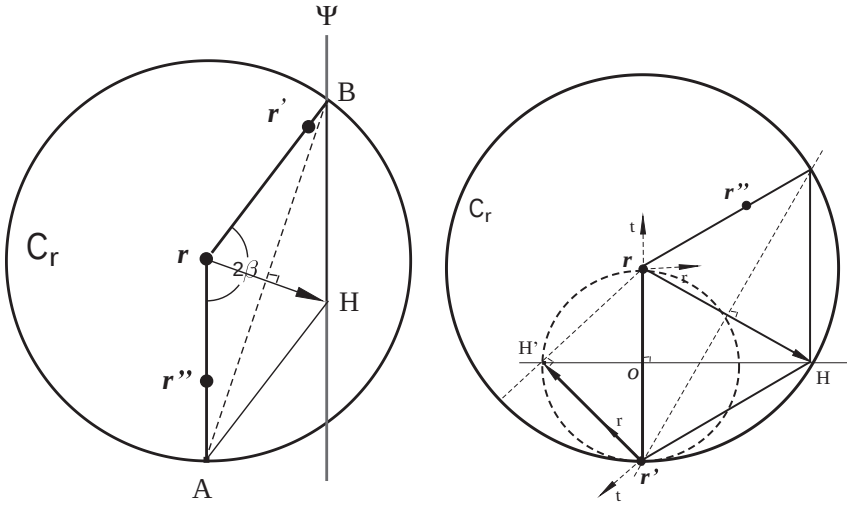
4 Flocchini et al. Algorithm [7] in the face of Eventual Compasses

In this section, we show that the algorithm proposed by Flocchini et al. [7] does not solve the gathering problem if we assume eventual compasses. The Flocchini et al. [7] algorithm is described briefly as follows. Recall that the distance graph is connected. Depending on the positions of the robots in its circle of visibility, robot r computes a destination point as follows:

- *Null move*: if r sees robots to its left or above on its vertical axis, it does not move.
- *Vertical move*: if r sees robots only below it on its vertical axis, it moves down toward the nearest robot.
- *Horizontal move*: if r sees robots only to its right, it moves horizontally toward the vertical axis of the nearest robot.
- *Diagonal move*: if r sees robots both below on its vertical axis and on its right, then it considers the nearest one below it and the nearest one to its right and performs a diagonal move as illustrated in Figure 2(a). In this move, the destination point of r is always computed in such away that it lies inside its circle of visibility. That is, r forms an angle 2β with its mutually visible robots, and this move ensures that the destination point is always computed based on an angle $60 \leq \beta \leq 90$ so that robot r remains visible to the robots in its visibility range.

Assume now that the robots will execute the Flocchini et al. [7] algorithm, but they do not agree on the compasses, that is the robots can have different directions and orientations of their x and y axes. As a result, the *distance graph* will be disconnected. We will demonstrate this by a counter example where the algorithm is unable to solve the problem.

Lemma 6 *In a system in which robots have eventual compasses, the algorithm of Flocchini et al. [7] results in a disconnection of the distance graph.*



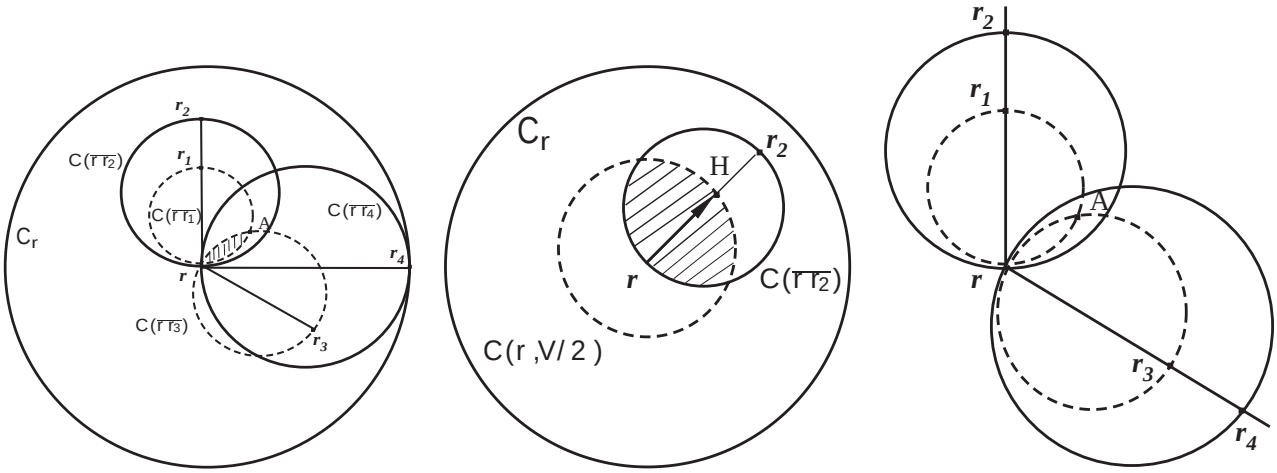
(a) Diagonal move in Flocchini et al. [7]

(b) r executes a diagonal move, while r' executes an horizontal move. As a result, $\overline{HH'} > V$: r and r' are not mutually visible any more.

Figure 2. Flocchini et al. [7] algorithm based on a perfect compass.

PROOF. Let us consider the example illustrated in Figure 2(b). Robot r performs a look operation $Look(r, t)$. According to its compass, r sees robot r' below it on its vertical axis and r'' to its right. By the algorithm, r will perform a *diagonal move*. Let H be its destination such that $H \in \mathcal{C}_r(t)$, i.e, $\beta = 60°$. Assume also that r' performs a look operation $Look(r', t)$. According to its compass, robot r' sees robot r to its right. Consequently, r' will perform an *horizontal move*. Let H' be the destination of r' such that, $\overline{rH'} = \overline{r'H'}$ as illustrated in Figure 2(b). By considering this simple example, we will show that the movement of r to H, respectively r' to H' will lead to a situation where r and r' are at distance greater than the visibility radius V; i.e, $\overline{HH'} > V$. Let us denote by $C(\overline{rr'})$, the circle with diameter $\overline{rr'}$, and by o, its center. $H' \in C(\overline{rr'})$ because $\overline{rH'} \perp \overline{r'H'}$ by the algorithm. We also have $H \in \mathcal{C}_r(t)$ because $\beta = 60°$ by construction. Then, $\overline{rH} = \overline{rr'} = \overline{r'H}$. Hence, the triangle $\triangle(r, r', H)$ is equilateral. As a result, the perpendicular on $\overline{rr'}$ passing through H passes by the center o of $C(\overline{rr'})$. In addition, the perpendicular to $\overline{rr'}$ passing through H', passes by o because the triangle $\triangle(r, r', H')$ is isosceles by construction. Hence, we compute the distance $\overline{HH'}$ as follows:

$$\begin{aligned}
\overline{HH'} &= \overline{oH} + \overline{oH'} \\
&= \sqrt{\overline{rH}^2 - \overline{or}^2} + \overline{oH'} \\
&= \sqrt{V^2 - (V/2)^2} + V/2 \\
&= \frac{\sqrt{3}}{2}V + V/2 > V
\end{aligned}$$



(a) r computes all the circles between itself and its mutually visible neighbors: r moves to A by at most $V/2$.

(b) r has only r_2 on its visibility circle: r moves toward r_2 by at most $V/2$.

(c) r_1 and r_3 are the two limit robots.

Figure 3. Principle of the algorithm.

Thus, we conclude that $\overline{HH'} > V$. In consequence, r and r' are unable to see each other any more. This results in a disconnection of the distance graph, and the lemma holds. $\square_{\text{Lemma 6}}$

Theorem 1 *In a system in which robots have eventual compasses, the algorithm of Flocchini et al. [7] does not solve the robots gathering problem.*

PROOF. From Lemma 2, the gathering problem is unsolvable if the distance graph is disconnected. However, from Lemma 6, we conclude that the algorithm of Flocchini et al. [7] leads to a disconnection of the distance graph if we assume eventual compasses. Hence, the lemma follows. $\square_{\text{Theorem 1}}$

5 Gathering with Eventual Compasses

Unlike the CORDA model, where the robots share *perfect compasses*, we assume that the robots have *eventual compasses*, which means that there is a time after which the agreement on the compass holds, and for a sufficiently long period, however this time is unknown to the robots. In this section, we present a deterministic algorithm for solving the gathering problem based on eventual compasses.

5.1 Description

The idea of the algorithm is to solve the problem by achieving the following two subgoals at every time instant t. (1) Robots that are mutually visible at time t remain mutually visible at time $t + 1$. (2) The robots located on

the leftmost side at time t "get closer" in some sense to their mutually visible robots (i.e, robots located on their visibility radius) at $t+1$.

The gathering algorithm is depicted in Algorithm 1 and described in more detail as follows. First, at every

Algorithm 1 Gathering with an Eventual Compass

Algorithm:

1: $compass_r$.query();
2: **if** $|\mathcal{U}| = 0$ **then**
3: Do_nothing(); {gathering terminated}
4: **else**
5: Ψ:= vertical axis passing through r according to $compass_r$;
6: **if** ($|Left_r| > 0$ **then**
7: Do_nothing();
8: **else if** $((|Top_r \bigcap \Psi|) > 0 \wedge |Right_r| = 0)$ **then**
9: Do_nothing();
10: **else**
11: S^{Ψ} := set of all robots on Ψ except r;
12: S^{R} := set of all visible robots to $Right_r$;
13: S:= $S^{\Psi} \bigcup S^{R}$;
14: **if** $|S| = 1$ **then**
15: $A := \{s \in S\}$
16: **else**
17: S^{robot} := Two_Limit_Robots(S); (see Function 2)
18: **for** (s in S^{robot}) **do**
19: $\{A, r\} := \bigcap_{s \in S^{robot}} C(\overline{rs})$;
20: **end for**
21: **end if**
22: **if** ($\overline{rA} \leq V/2$) **then**
23: Move(A);
24: **else**
25: $H := \{(p \in \overline{rA})/\overline{rp} = V/2\}$;
26: Move(H);
27: **end if**
28: **end if**
29: **end if**

time instant t where robot r becomes active, it query its compass to get information on the direction and then, it considers all the robots in its circle of visibility $\mathcal{C}_r(t)$. If robot r sees robots to its left side $Left_r$, then it does not move. This restriction will allow the robots located on the leftmost side to move toward the ones located on the right, and hence, fulfill the second subgoal. If robot r sees only robots above it on its vertical axis Ψ and no robots on its right, then it does not move; due to the asynchrony of the robots, this restriction is required in order to break symmetry. For instance, in a system with two robots that occupy distinct positions, it might happen that each time, each robot would try to move to the position of the other; as a result, the two robots would swap positions endlessly, and the algorithm would never terminate. Let us now consider that robot r has robots to its right, and maybe some above it on its vertical axis, but none to its left. Then, with each robot in its circle of visibility, r computes the circle with the diameter the two points corresponding, respectively, to its position and the position of its visible neighbor r' (i.e, $C(\overline{rr'})$). S is the set of robots that r sees on its circle of visibility $\mathcal{C}_r$ at time t, and C^S is the set of all the circles that r forms with its neighbors in C^S (refer to Figure 3(a)). To achieve the first subgoal, i.e, keep the sight with the robots on its circle of visibility, r must move to one of the points in the intersection of

all the circles in C^S. This point must be at a distance of at most $V/2$ from its current position. This move ensures that r always stays mutually visible with its neighbors. Depending on the intersection of the set of circles C^S, the destination of r is computed as follows:

- If the intersection of the set C^S is one point, then obviously this point is r. In this case, the destination point of r is itself. Hence, r performs a *null move*.
- $C^S = 1$, that is, r sees only one robot on its visibility circle (for instance, r_2 in Figure 3(b)). In this case, r moves toward r_2 by $V/2$ if $\overline{rr_2} > V/2$; otherwise it moves to r_2.
- The intersection of the set C^S is a *region* (refer to Figure 3(a)). In this case, r can move to any point in this region by a distance of at most $V/2$. Notice that, this region always results from the intersection of exactly two circles in C^S. These two circles are the circles that r forms with exactly two robots; we refer to these robots as the *two limit robots* (they are given by Function 2). The intersection of these two circles is two points, one point is r, and the second point, say A. The destination of r is A if the distance $\overline{rA}$ is less than or equal to $V/2$; otherwise, it is the point H that belongs to the segment $\overline{rA}$, and at a distance $V/2$ from r.

Note that, by the algorithm, the maximum distance that a robot can travel in one cycle is at most $V/2$. We now explain how the two limit robots are computed. They are given by the function *Two_limit_robots(S)* which is depicted in Algorithm 2. This function takes as input the set of all the robots in the visibility circle of robot r, and returns as result a set of exactly two robots; these robots constitutes the two limit robots. Given the set S of robots in the visibility circle of r, r computes the list of the angles that it forms with each pair of robots in S. Note that these angles have r as a corner. The two limit robots are the robots that form the maximum angle with r. If there is more than one pair of robots that forms the maximum angle with r, then among these robots, some of them are aligned together and aligned with r. In this case, the robot with minimal distance from r is selected (refer to Figure 3(c)).

Algorithm 2 Function Two_Limit_robots(S)

Function Two_Limit_robots(S)

1: $S^{cand} := \{r_1, r_2\} / \angle r_1 r r_2 = MAX\{\angle r' r r'', \forall r', r'' \in S \wedge (r' \neq r'')\}$;
2: **if** ($|S^{cand}| = 2$) **then**
3: Return (S^{cand})
4: **else**
5: $S^{res} := \varnothing$;
6: $S_1 = \{r_i \in S^{cand} / (r, r_i) \in wedge1\}$
7: $S_2 = \{r_j \in S^{cand} / (r, r_j) \in wedge2\}$
8: **if** $\overline{rr_1} = MIN\{\overline{rr'} / r' \in S_1\}$ **then**
9: $S^{res} := S^{res} \bigcup \{r_1\}$;
10: **end if**
11: **if** $\overline{rr_2} = MIN\{\overline{rr''} / r'' \in S_2\}$ **then**
12: $S^{res} := S^{res} \bigcup \{r_2\}$;
13: **end if**
14: Return (S^{res})
15: **end if**

6 Correctness

In this section, we prove the correctness of our algorithm by presenting its proof of correctness. We proceed in two steps to prove that our algorithm is correct. In the first step, we show that the connectivity of the distance graph is preserved during the entire execution of the algorithm. That is, regardless of the orientation of their compasses, the robots which are initially visible remain always visible. In a second step, we show that after the time GST, if the agreement on the compasses of the robots holds for a long enough time, then all robots will gather at one point in finite number of steps. Before proceeding, let us recall an important lemma proved by Flocchini et al. [7]; if the *distance graph* is disconnected, the gathering problem is unsolvable. Hence, in the following, we will always assume that the distance graph is connected.

6.1 Preserved connectivity

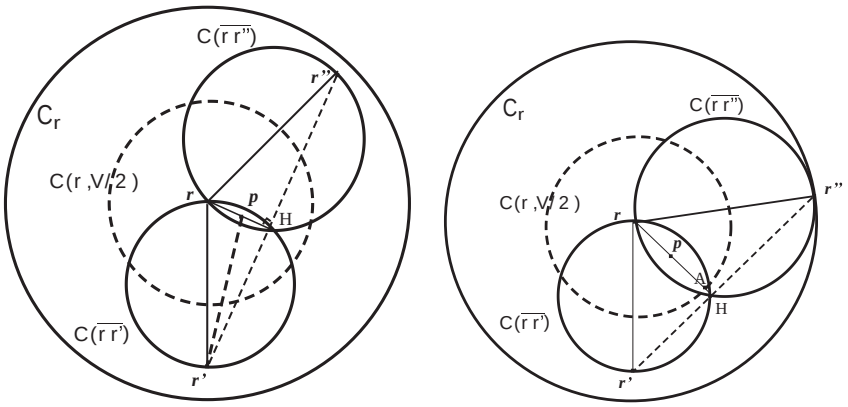
We now prove that the connectivity of the distance graph is preserved during the entire execution of the algorithm. The following two lemmas show that the robots which are initially mutually visible remains mutually visible during the entire execution of the algorithm until they collide on the same point. Recall that the robots may disagree on the orientations of their compasses, including the robots that are located on the same point.

Lemma 7 *Let $\mathcal{U}$ be composed of two robots r and r' that are mutually visible at time t. Then, at any time $\bar{t} > t$, r and r' are mutually visible or collide at the same point. Moreover, at all times between t and $\bar{t}$, r and r' are at a distance at most V.*

PROOF. Recall that the compasses of the robots may disagree. Consequently, three cases may arise. In the first case, both r and r' do not move. In the second case, only one robot decides to move. In the third case, both r and r' move. The three cases are detailed as follows:

- **Both r and r' do not move**. Observe that the lemma trivially holds if $Left_r$ is not empty when r performs $Look(r,t)$ (r sees r' to its left) and also $Left_{r'}$ is not empty when r' performs $Look(r',t')$ (r' also sees r to its left) with $t' \geq t$. In fact, according to the algorithm none of the robots moves. Thus, r and r' remain at the initial distance from each other which is less than V. Similarly, the lemma holds if $Top_r \bigcap \Psi$ is not empty as a consequence of $Look(r,t)$ (r sees r' above it on its vertical axis) and $Top_{r'} \bigcap \Psi$ is not empty as a consequence of $Look(r',t')$. Again, according to the algorithm none of the robots moves.

- **Only one robot moves**. Assume that $Left_r$ is empty when r performs $Look(r,t)$ and $Left_{r'}$ is not empty when r' performs $Look(r',t')$, with $t' \geq t$ that is r' sees r to its left. Consequently, r' can not move as long as it sees r to its left. Let H be the destination point computed by r as a consequence of its look operation $Look(r,t)$. Then, by the algorithm r moves to r' if $\overline{r(t)r'(t)} \leq V/2$ (in this case $H = r'$), otherwise it moves towards r' by $V/2$ ($\overline{rH} = V/2$), thus shortening their distance. Assume that r finishes its move at time $t'' > t$. Between time t and t'', r will be to the left of r', hence r' can not move. Thus, $\forall p \in \overline{r(t)r'(t)}, \overline{pr'(t)} \leq V$. By applying the same argument, the lemma holds if $Top_r \bigcap \Psi$ is empty when r performs $Look(r,t)$ and $Top_{r'} \bigcap \Psi$ is not empty when r' performs $Look(r',t')$ with $t' \geq t$. Similarly, r will move toward r', and $\forall q \in \overline{r(t)r'(t)}, \overline{qr'(t)} \leq V$.



(a) r computes as destination H: case where $\overline{rH} \leq V/2$.

(b) r computes as destination H: case where $\overline{rH} > V/2$.

Figure 4. The movement of robot r preserve the vision connection with r' and r''.

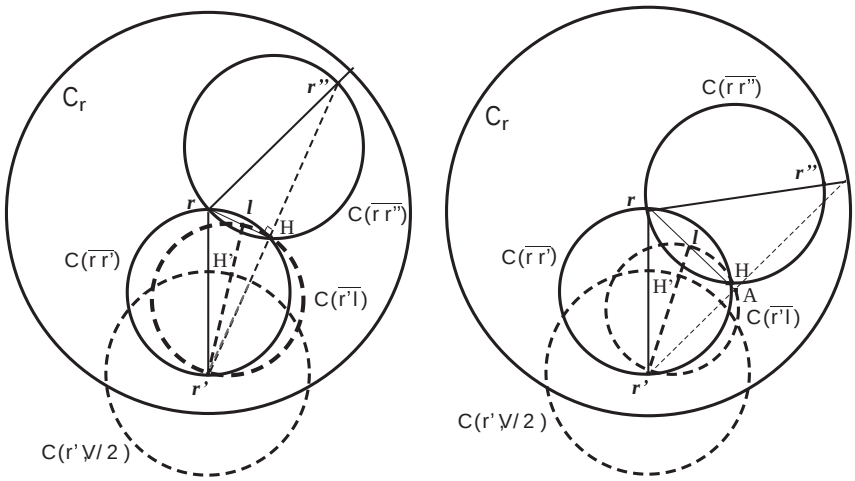
- **Both r and r' move**. This case is interesting because both robots r and r' move. Assume that for r, $Right_r$ is not empty when r performs $Look(r,t)$. Similarly, for r', $Right_{r'}$ is not empty when r' performs $Look(r',t')$. Let H be the destination point computed by r as a consequence of its look operation $Look(r,t)$. Let us first assume that $\overline{r(t)r'(t)} \leq V/2$. That is $H = r'(t)$. Thus, r will move to the point $r'(t)$, and consequently $\forall p \in \overline{r(t)r'(t)}, \overline{pr'(t)} \leq V/2 \leq V$. Let also H' be the destination point computed by r' as a consequence of its look operation $Look(r',t')$ with $t' \geq t$. Then, by the algorithm H' is equal to the position of r at time t' ($H' = r(t')$). Consequently, $\forall q \in \overline{r(t')r'}, \overline{r(t')q} \leq V/2 \leq V$. In addition, $\forall p \in \overline{r(t)r'(t)}, \forall q \in \overline{r(t')r'(t')}, \overline{pq} \leq V/2 \leq V$.

  Let us now assume that $\overline{r(t)r'(t)} > V/2$, Then, according to the algorithm r moves toward $r'(t)$ by $V/2$ (let's call this point H). Consequently $\forall p \in \overline{r(t)H}, \overline{pr'(t)} \leq V$ since $\overline{r(t)r'(t)} \leq V$. Let also H' be the destination point computed by r' as a consequence of its look operation $Look(r',t')$ with $t' \geq t$. By the algorithm, if $\overline{r'(t')r(t')} \leq V/2$. Then, r' moves to $r(t')$, otherwise r' moves toward $r(t')$ by $V/2$. Consequently, $\forall q \in \overline{r'(t')H'}, \overline{qH'} \leq V/2$, and thus $\overline{qr(t')} \leq V/2 \leq V$. In addition, $\forall p \in \overline{r(t)r'(t)}, \forall q \in \overline{r'(t')r(t')}, \overline{pq} \leq V$, and the lemma holds.

□Lemma 7

Lemma 8 *Let the pairs of robots r and r' and r and r'' be mutually visible at time t. Moreover, let r' and r'' be the two limit robots of r at time t. Then, at any time $\bar{t} > t$, r and r' and r and r'' are mutually visible or collide on the same point. Moreover, at all times between t and $\bar{t}$, r and r' and r and r'' are at a distance at most V.*

PROOF. The proof consists of showing that the movement of any of the robots at any time will not lead to a situation in which the pairs of robots r and r' and r and r'' are at distance greater than V from each other. Depending on the orientation of the compass of r, r' and r'', we shall consider three possible situations:



(a) r computes as destination H. $\overline{rH} \leq V/2$ (H belong to $\overline{r'r''}$). r' sees r while moving at position l. The destination of r' lies inside $\mathcal{C}(\overline{r'l})$.

(b) r computes as destination H. $\overline{rH} > V/2$ (H does not belong to $\overline{r'r''}$). r' sees r while moving at position l. The destination of r' lies inside $\mathcal{C}(\overline{r'l})$.

Figure 5. The movement of robot r and r' preserve the vision connection of the robots.

- **Only r moves (r', r'' do not move)**. r performs a look operation $Look(r,t)$ and decides to move. Let t' and t'' the time when r starts its move and when it stops, respectively. r' and r'', whenever they look between t' and t'', they decide not to move (because they see some robots to their left, or each of them computes as a destination point its current location.) Let H be the destination point computed by r as a consequence of its look operation at time t. By the algorithm, $H \in C(\overline{rr'})$ and $H \in C(\overline{rr''})$.

  Let us first assume that $\overline{rH} \leq V/2$ (refer to Figure 4(a)). By Lemma 4 on triangle $\triangle(r, H, r')$, $\forall p \in \overline{rH}$, $\overline{pr'} \leq \overline{rr'} \leq V$. Similarly, by Lemma 4 on triangle $\triangle(r, H, r'')$, $\forall q \in \overline{rH}$, $\overline{qr''} \leq \overline{rr''} \leq V$.

  Let us now assume that $\overline{rH} > V/2$ (refer to Figure 4(b)). Let A be the point that belongs to $\overline{rH}$ such that $\overline{rA} = V/2$. Since $r, H \in C(\overline{rr'})$, then by Lemma 3, A lies inside the circle $C(\overline{rr'})$. It follows that by Lemma 2, $\forall p \in \overline{rA}$, $\overline{pr'} \leq \overline{rr'} \leq V$. By applying the same argument on r'', we get $\forall q \in \overline{rA}$, $\overline{qr''} \leq \overline{rr''} \leq V$. Consequently, as long as r' and r'' do not move, the distance between r and r', and r and r'' is at most V between the time t' and t''.

- **Both r and r' move (r'' does not move)**. Robot r performs a look operation $Look(r,t)$ and decides to move. Let H be its destination point (refer to Figure 5(a)). Let us first assume that $\overline{rH} \leq V/2$. We denote by t' and t'' the time when r starts its move, and when it stops, respectively. r' also decides to move as a consequence of some of its look operations performed between time t' and t''. We denote by $\bar{t}$ the time when r' performs its look operation whose result is a real movement ($t' < \bar{t} \leq t''$). Let H' be its computed destination. Let l be the point position of r at time $\bar{t}$. Explicitly, r' observes r at time $\bar{t}$ at position l. We denote by $\bar{t'}$ and $\bar{t''}$ the time when r' starts its move, and when it stops, respectively. Assume that $\bar{t''} \geq t''$. Now we will show that H and H' are at distance at most V. Moreover, at all times between , t' and $\bar{t''}$, r and

r' are at distance at most V. Notice that, for any time between t' and $\bar{t'}$, r and r' are at distance at most V. This was demonstrated in the above case (before time $\bar{t'}$, r' didn't start to move yet). Between time $\bar{t'}$ and $\bar{t''}$, the proof consists of showing that $\forall p \in \overline{lH}$, and $\forall q \in \overline{r'H'}$, $\overline{pq} \leq V$. As r' includes r on its computation at position l, then, the destination H' of r' is at least a point in the intersection of the two circles $C(\overline{r'l})$ and $C(r', V/2)$. Therefore, H' lies inside $C(\overline{r'l})$. By Lemma 5, H belongs to $\overline{r'r''}$, thus $\overline{rH} \perp \overline{Hr'}$. We have also $l \in \overline{rH}$, therefore $\overline{lH} \perp \overline{Hr'}$. Hence, $H \in C(\overline{lr'})$. As a consequence, $\forall p \in \overline{lH}$, l is inside $C(\overline{r'l})$. In addition, $\forall q \in \overline{r'H'}$, q is inside $C(\overline{r'l})$, and then by Lemma 2, $\forall p \in \overline{lH}$, $\forall q \in \overline{r'H'}$, $\overline{pq} \leq \overline{lr'} \leq V$.

Let us consider now the case where the distance between r and its computed destination is greater than $V/2$ (refer to Figure 5(b)). Let H be the point at distance $V/2$ from r, and A be the second point of intersection of $C(\overline{rr'})$ and $C(\overline{rr''})$. Observe that $l \in \overline{rA}$, and $\overline{rA} \perp \overline{r'A}$. Then, $A \in C(\overline{r'l})$. Thus, $\forall p \in lA$, p is inside $C(\overline{r'l})$. Therefore, H is inside $C(\overline{r'l})$. Consequently, $\forall p \in \overline{lH}$, p is inside $C(\overline{r'l})$. In addition, $\forall q \in \overline{r'H'}$, q is inside $C(\overline{r'l})$, and then by Lemma 2, it follows that $\forall p \in \overline{lH}$, $\forall q \in \overline{r'H'}$, $\overline{pq} \leq \overline{lr'} \leq V$.

- **All robots r, r' and r'' move**. Robot r performs a look operation $Look(r, t)$ and decides to move. We denote by t' and t'' the time when r starts its move, and when it stops, respectively. r' and r'' both decides to move as a consequence of some of their look operations performed between time t' and t''. The proof for this case is the same as the previous one, and is thus omitted here.

In the three cases, r and r' and r and r'' are at a distance of at most V. Moreover, at all times during their movements, r and r' and r and r'' are at a distance of at most V. This completes the proof. $\square_{\text{Lemma 8}}$

The above two lemmas show that if two robots are mutually visible, they will continue to be so at future times.

Lemma 9 *Let t_0, be the time when the robots are in their initial configuration. Then, at any time $\bar{t} > t_0$, if all the robots gather or not on the same point, the distance graph is connected.*

PROOF. We omit this proof since it has been proved in [7]. $\square_{\text{Lemma 9}}$

From Lemma 7, Lemma 8, and Lemma 9, we conclude that:

Theorem 2 *The distance graph is connected during the entire execution of Algorithm 1.*

6.2 Termination of the algorithm

In this section, we show that the robots will gather at one point in finite number of steps. To do so, we assume that, the robots reach the time GST, and their compasses agree for a sufficiently long time after GST. First of all, we show that in any configuration with two robots at distinct positions, in finite number of steps, all the robots gather at one point in finite time. Then, we show that any configuration, with three or more robots in distinct positions in reduced to a configuration of two robots at distinct positions in finite time.

Lemma 10 *Let $\mathcal{U}$ be composed of two robots r and r' that are mutually visible at time t. Then, there exists a time $\bar{t} > t$ where r and r' collide at the same point.*

PROOF. Let us assume that r' is prevented to move because of r, then r is either located to the left of r' or above r' on its vertical axis. Let us consider the first case where r is on the left of r'. Then, according to the

algorithm, r' is prevented to move as long as it sees r to its left. While r is moving, it will be always to the left of r'. Hence, r' will stay still. By Assumption 1, the minimum distance travelled by a robot in a move is δ_r, then the number of steps performed by r to reach r' is at most $\overline{rr'}/\delta_r$ which is finite. Hence, r reaches r' in finite time. Similarly, if r lies above r' on its vertical axis, then by the algorithm, r' can not move as long as it sees r' on its top. Then, by the same argument as above, r will reach r' in finite number of steps and the lemma holds. $\square_{\text{Lemma 10}}$

Lemma 11 By the algorithm, if a robot r sees robots both below and above it on its vertical axis Ψ, it will execute a null movement.

PROOF. Let r_1 and r_2, be two robots above r on its vertical axis Ψ, and r_3 is below r on Ψ. By the algorithm, the intersection of the circles $C(\overline{rr_1})$, $C(\overline{rr_2})$ and $C(\overline{rr_3})$ is equal to r. In addition, the intersection of these circles with the circle $C(r, V/2)$ is r. Consequently, r computes as destination its current position r. Then, r will stay still, and the lemma holds. $\square_{\text{Lemma 11}}$

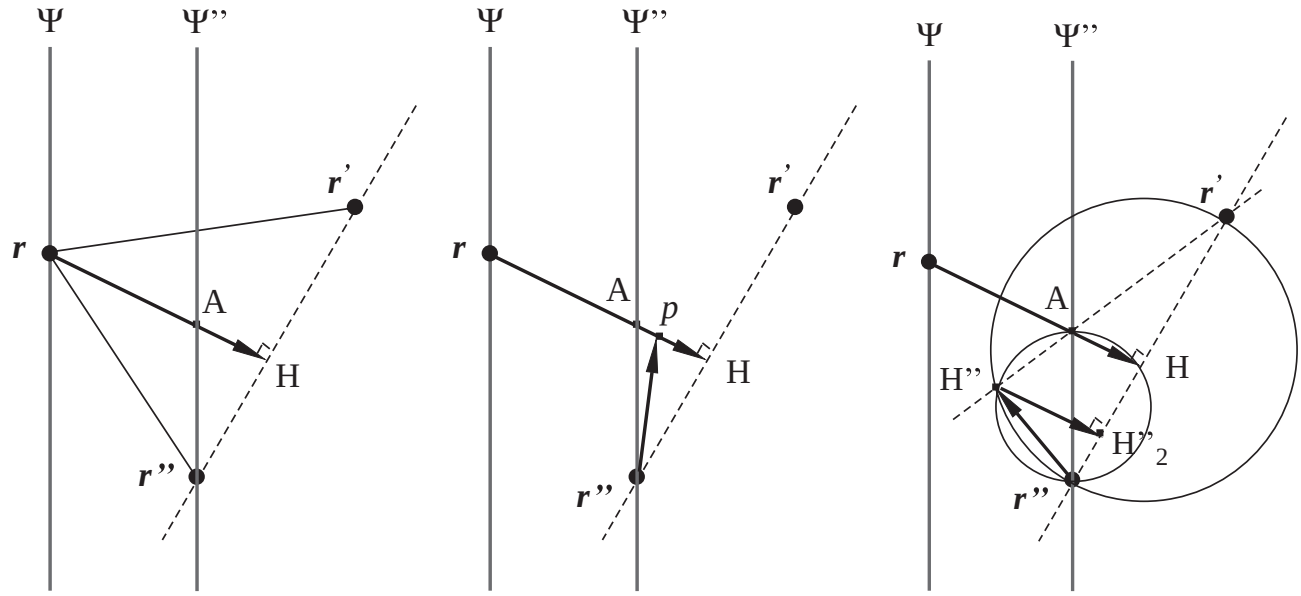
Lemma 12 *Any configuration with three or more robots that are located at distinct positions, and collinear is reduced to a configuration with two robots at distinct positions in a finite number of steps.*

PROOF. In a configuration where robots are collinear, there exist two cases; either all the robots are located on the same vertical axis Ψ, or all the robots are collinear, but not on Ψ. In both cases, exactly one robot can move at any time instant. Consider the first case, only the topmost robot can move at a time instant t. In addition, by Assumption 2, the cycle of a robot is finite and by Assumption 1, the distance travelled by a robot in a move is at least δ_r. Then, in any configuration, the topmost robot at time instant t will gather with the robot below it in finite time. Since the number of robots is finite, then, in finite time this configuration is reduced to a configuration of two robots at distinct positions. In the second case, by using the same argument as above, in any configuration, the leftmost robot will reach the nearest one to its right in finite time. Since the number of robots is finite, then, this configuration is reduced in finite time to a configuration of two robots at distinct positions. In both cases, any configuration in which three or more robots are collinear is reduced to a configuration of two robots in finite time, and the lemma holds. $\square_{\text{Lemma 12}}$

Lemma 13 *Any configuration with three or more robots located at distinct positions and non collinear is reduced to a configuration with two robots at distinct positions in finite number of steps.*

PROOF. We show that, at any time instant, in any configuration of three robots, one or two robots are able to move, and that their movements will lead in finite time to a configuration where only one robot can move. From this configuration, there is also a finite time within which this configuration is transformed into configuration of two robots at distinct positions.

Let r, r' and r'' be three robots located at distinct positions with r and r', respectively, r and r'' are mutually visible at time t. Let r lie on $Left_{r'}$ and $Left_{r''}$ at time t. By the algorithm, r' and r'' are unable to move at time t. Let H be the destination point computed by r as a consequence of its look operation at time t. By Lemma 5,



(a) Robot r computes H as destination. While r is moving to H, r' and r'' see r at position q to the right of A. Hence, both r' and r'' do not move.

(b) r moves to H. r'' sees r at position p and moves to p.

(c) r moves to H. r'' sees r at position A and computes as destination H''.

Figure 6. Termination of the algorithm: 3 robots

H is collinear with r' and r''. Let r'' be closer to r in distance than r' at time t. We denote by Ψ'', the vertical axis passing through r'' and by A, the intersection of Ψ'' with the segment $\overline{rH}$ (refer to Figure 6(a)). Let $t' > t$ and $t'' > t'$ be the time when r starts its move, and when it stops, respectively. Let us assume that $\overline{rH} \leq V/2$ (the case where $\overline{rH} > V/2$ is the same, thus omitted here). By definition, r is to the left of r' for any position q that belongs to $\overline{rH}$. As a consequence, robot r' will not perform any movement between t' and t''. Now consider r'', depending where r'' sees robot r while it is moving, and whether r' and r'' are mutually visible at time t or not, we distinguish the following cases. Let us first assume that r' and r'' are not mutually visible at time t.

- r'' **observes** r **at position** q **before** A **or on** A (refer to Figure 6(a)). Consider robot r moving toward H, by definition, it is to the left of r'' for any position q that belongs to $\overline{rA}$ and $q \neq A$. In consequence, r'' does not move. Assume now that r reaches the point A, then, it becomes above r'' on its vertical axis Ψ''. In addition, r'' can not see any robot to its right by hypothesis (r' and r'' are not mutually visible). Again, by the algorithm, r'' will not move.

- r'' **observes** r **at position** p **to the right of** A (refer to Figure 6(b)). Assume that robot r reaches a position p which is to the right of A at time $\bar{t} > t'$. Assume also that r'' performs a look operation at time $\bar{t''} \geq \bar{t}$, and it sees r at position p, which is to its right. Then, by the algorithm, r'' will move to p by $V/2$. Assume also that $\overline{pr''} \leq V/2$. Then, after finishing their movements, robot r and r'' will be respectively at positions H and p. Robot r' remains stationary. Let us call this configuration γ. Both robots r and r'' reach the

configuration γ in a finite time; the maximum number of steps performed by r to reach H is $\overline{rH}/\delta_r$ which is finite. Also, the maximum number of steps performed by r'' to reach p is $\overline{r''p}/\delta_r$ which also finite. In this new configuration γ, r'' and r' become mutually visible. We will now show that this configuration γ is also transformed in finite time into a configuration γ' with two robots at distinct positions. In γ, r'' is located to the left of r (at position H) and r'. Then, both r and r' can not perform any movement. For robot r'', it includes both r and r' in its computation. Let H'' be its computed destination. By the algorithm, H'' is collocated with r at H. Then, r'' moves to the position of r. This movement is done in finite time. Hence, r and r'' gather in the same point H in finite time. As a result, the configuration γ with three robots at distinct positions is transformed into configuration γ' with two robots at distinct positions in finite time.

- r'' **observes** r **at position** H. If robot r'' observes r only at position H, then this configuration represents a collinear configuration of three robots at distinct positions, and by Lemma 12 , it is reduced in finite time to a configuration of two robots at distinct positions (in this case r'' will gather with r at position H).

Let us now assume that r' and r'' are also mutually visible at time t. Again, r' will not perform any movement because it sees robot r and r'' to its left. Consider now r''. If r'' observes r at position q before A, then in this case r'' will not move because r is located to its left. If r'' observes r at position H, then this case also represents a collinear configuration of three robots at distinct positions, and by Lemma 12 it is reduced in finite time to a configuration of two robots at distinct positions. Let us now consider the more interesting case where r'' observes r at position A or at position p to the right of A (refer to Figure 6(c)). We will only treat this case in which r'' observes r at position A. The case in which r'' observes r at position p to the right of A is very similar and is therefore omitted here. Let H'' be the destination of r''. By Lemma 5, H'', A and r' are collinear. From Assumption 2, the cycle time of a robot is finite, hence, respectively r and r'' reach H and H'' in finite time. Let us call this configuration γ. In this configuration, both r and r' are unable to move because r'' is located to their left (at position H''). Let H_2'', be the destination computed by r'' as a consequence of its look operation performed in γ. By Lemma 5, H_2'' is collinear with r (position H) and r'. Furthermore, it is located to their left. Thus, for any position q that belongs to $\overline{H''H_2''}$, r'' is to the left of r and r'. Again neither robot r nor r' can move until r'' reaches H_2''. As a consequence, the movement of r'' to H_2'' results on a collinear configuration, and from Lemma 12 , this configuration is reduced to a configuration of two robots in finite time.

In all cases, any configuration with three or more robots located at distinct positions is reduced to a configuration where only two robots are at distinct positions in finite time, and the lemma holds. $\square_{\text{Lemma 13}}$

Lemma 14 *Any configuration in which three or more robots are located at distinct positions is reduced to a configuration with two robots at distinct positions in finite number of steps.*

PROOF. From Lemma 12, any configuration in which three or more robots are collinear is reduced to a configuration of two robots at distinct positions in finite time. By Lemma 13, we conclude that any configuration with three or more robots at distinct positions and non collinear is reduced to a configuration with two robots at distinct positions in finite time, hence, the lemma follows. $\square_{\text{Lemma 14}}$

Lemma 15 *Under Algorithm 1, all the configurations in which all the robots gather in one point are stable.*

PROOF. Assume that at some time t, all the robots gather in one point. In such a configuration, none of the robots sees other robots on its visibility circle. Thus, by the algorithm, none of the robots will ever move. Consequently, such configuration is stable and the lemma holds. $\square_{\text{Lemma 15}}$

Theorem 3 *Under Algorithm 1, all the robots gather at one point in finite time.*

PROOF. From Lemma 14, any configuration with three or more robots at distinct positions is reduced to a configuration with two robots at distinct positions in finite time. By Lemma 10, any configuration of two robots located at distinct positions will lead to a configuration wherein the two robots gather at one point in finite time. Moreover, by Lemma 15, this configuration is stable. This terminates the proof. $\square_{\text{Theorem 3}}$
By Theorem 2 and Theorem 3, it follows that:

Theorem 4 *In an asynchronous system, with n anonymous, oblivious mobile robots, with limited visibility, and assuming eventual compasses, the gathering problem is solvable deterministically.*

7 Conclusion

In this paper, we took a new look at the gathering of a group of oblivious asynchronous mobile robots with limited visibility. While, previous work [7] showed that this problem can be solved if the robots are equipped with compasses, we have shown that their algorithm can not tolerate any transient failure of those compasses. We have shown that gathering can nevertheless be solved with unreliable compasses, as long as these compasses eventually stabilize to provide consistent information to the robots (the robots are not aware of when this actually happens). The main benefit of our algorithm is that we solve the problem with a weaker assumption by assuming eventual compasses rather than perfect ones. This weaker assumption leads to stronger results. In particular, it allows the algorithm to tolerate any number of transient failures, and also it gives the algorithm the nice property of self-stabilization. One important result of this work is that a perfect compass is unnecessary to maintain vision connectivity between robots. However, it is only required for the termination of the algorithm (i.e, gather the robots at a single point in finite time). For this reason, we can claim that one property, "*eventual compass*", has the same computational power as "*perfect compass*" for solving the robot gathering problem if the agreement on the compasses of robots eventually holds for a sufficiently long period. This result is important in practice. Finally, the results of this paper raise new interesting research questions. For instance, the eventual compasses constitutes one class of unreliable compass. An other class of unreliable compasses is the *imprecise compasses*. The question is can we still solve the gathering problem if the compasses of the robots are imprecise, i.e, they are bounded by some errors. Currently, we are investigating this issue.

Acknowledgments

We are especially grateful to Hirotaka Ono for his insightful comments regarding this work.

References

[1] N. Agmon and D. Peleg. Fault-tolerant gathering algorithms for autonomous mobile robots. In *Proc. of the Fifteenth Annual ACM-SIAM Symposium on Discrete algorithms (SODA'04)*, pages 1070–1078, Philadelphia, PA, USA, 2004.

[2] 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.

[3] M. Cieliebak. Gathering non-oblivious mobile robots. In *Proc. of the 6th Latin American Symposium on Theoretical Informatics (LATIN'04)*, pages 577–588, Buenos Aires, Argentina, May 2004.

[4] M. Cieliebak, P.Flocchini, G. Prencipe, and N. Santoro. Solving the robots gathering problem. In *Proc. of International Colloquium on Automata, Languages andProgramming ICALP'03*, pages 1181–1196, Eindhoven, Netherlands, 2003.

[5] R. Cohen and D. Peleg. Robot convergence via center-of-gravity algorithms. In *Proc. of Colloquium on Structural Information and Communication Complexity (SIROCCO'04)*, number 3104, pages 79–88, 2004.

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

[7] P. Flocchini, G. Prencipe, N. Santoro, and P. Widmayer. Gathering of asynchronous robots with limited visibility. In *Journal of Theoretical Computer Science*, 2005.

[8] G. Prencipe. Instantaneous actions vs. full asynchronicity: Controlling and coordinating a set of autonomous mobile robots. In *Proc. 7th Italian Conf. on Theoretical Computer Science (ICTCS'01)*, pages 154–171, Torino, Italy, 2001.

[9] G. Prencipe. CORDA: Distributed coordination of a set of autonomous mobile robots. In *Proc. 4th European Research Seminar on Advances in Distributed Systems (ERSADS'01)*, pages 185–190, Bertinoro, Italy, May 2001.

[10] G. Prencipe. On the feasibility of gathering by autonomous mobile robots. In *Proc. of Colloquium on Structural Information and Communication Complexity SIROCCO'05*, pages 246–261, 2005.

[11] M. Schneider. Self-stabilization.

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