Stabilizing flocking via leader election in robot networks

Davide Canepa, Maria Gradinariu Potop-Butucaru

▶ **To cite this version:**

Davide Canepa, Maria Gradinariu Potop-Butucaru. Stabilizing flocking via leader election in robot networks. [Research Report] RR-6268, INRIA. 2007, pp.29. inria-00166630v2

HAL Id: inria-00166630

https://inria.hal.science/inria-00166630v2

Submitted on 8 Aug 2007

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

INRIA

INSTITUT NATIONAL DE RECHERCHE EN INFORMATIQUE ET EN AUTOMATIQUE

Stabilizing flocking via leader election in robot networks

Davide Canepa — Maria Gradinariu Potop-Butucaru
Université Pierre et Marie Curie (Paris 6), CNRS,INRIA, France

canepa.davide@tiscali.it, maria.gradinariu@lip6.fr

N° 6268

Jully 2007

Thème COM

ISSN 0249-6399 ISRN INRIA/RR--6268--FR+ENG

INRIA
ROCQUENCOURT

Stabilizing flocking via leader election in robot networks

Davide Canepa , Maria Gradinariu Potop-Butucaru
Universit Pierre et Marie Curie (Paris 6), CNRS,INRIA, France
canepa.davide@tiscali.it, maria.gradinariu@lip6.fr

Thème COM — Systèmes communicants
Projet REGAL

Rapport de recherche n° 6268 — Jully 2007 — 26 pages

Abstract: Flocking is the ability of a group of robots to follow a leader or head whenever it moves in the plane (two dimensional Cartesian space). In this paper we propose and prove correct an architecture for a self-organizing and stabilizing flocking system. Contrary to the existing work on this topic our flocking architecture do not relies on the existence of a specific leader *a priori* known to every node in the network. In our approach robots are uniform, start in a arbitrary configuration and the head of the group is democratically elected via algorithmic tools.

Our architecture includes three modules: a leader election module, a preprocessing module and a motion module. The leader election module returns to each robot its status : leader or follower. The preprocessing module aims to arrange robots in a moving formation. The motion module provides the rules that will make robots change their positions whenever the leader moves. Every modification of robots position will be done maintaining alive the moving formation. For each of these modules we propose deterministic or probabilistic algorithms (in the case when a deterministic solution is impossible). Moreover, we prove their correctness and compute their complexity.

Our contribution is three fold. We propose novel deterministic and probabilistic solutions for leader election when robots evoluate in an asynchronous environment. We also prove the impossibility of deterministic leader election when robots have no common coordinates and start in an arbitrary configuration.

Secondly, we propose a collision free deterministic algorithm for circles formation designed for asynchronous networks.

Thirdly, we propose a deterministic flocking algorithm totally independent of the existence of *a priori* known leader. The proposed algorithm also works in asynchronous networks and does not assume common coordinates.

Key-words: Robot networks, Leader election, Flocking, Cercle formation

Unité de recherche INRIA Rocquencourt
Domaine de Voluceau, Rocquencourt, BP 105, 78153 Le Chesnay Cedex (France)
Téléphone : +33 1 39 63 55 11 — Télécopie : +33 1 39 63 53 30

Election de leader et déplacement de robots dans le plan

Résumé : L'élection de leader est une des briques de base dans la construction des systèmes réparties. Dans ce papier nous montrons que l'élection déterministe de leader est impossible dans un réseau de robots oblivious et proposons des algorithmes probabilistes pour contourner ce résultat d'impossibilité. De plus, nous proposons des algorithmes efficace de formation de cercle et déplacement de robots construitent au dessus d'un service d'élection de leader.

Mots-clés : Reseau de robots, Election de leader, Formation de cercle

1 Introduction

Several applications like large-scale construction, hazardous waste cleanup, space missions or exploration of dangerous or contaminated area motivate the research related to self-organized robot networks (multi-robot systems). The literature proposed so far a significant amount of research towards the operation of single remote robot, however more work is required towards the operation of networks of autonomous robots. These systems provide interesting solutions to many real problems: manipulation of large objects, system redundancy, reducing time complexity for the targeted tasks, however they bring in discussion some specific difficulties. In particular, these robots should achieve their tasks without human intervention based only on the information provided by the robots in the same group. Moreover, they have to explore unknown or quasi unknown environments while avoiding collisions among themselves. Additionally, they have to be able to reorganize whenever one or more robots in the group stop to behave correctly.

In this paper we propose a self-organized and stabilizing flocking architecture. Flocking is the ability of a group of robots to follow a leader or a flock-head whenever this one changes its position in plane. Our work is developed in CORDA model [1, 2] one of the two theoretical models proposed so far for robot networks. The first model proposed in the literature was introduced by Suzuky and Yamashita [3, 4, 5]. In this model robots are oblivious and perform a cycles of elementary actions as follows : observation (the robot observes the environment), computation (the robot computes its next position based on the information collected in the observation phase) and motion (the robot change its position to the coordinates returned by the computation phase). In this model robots cannot be interrupted during the execution of a cycle. The CORDA model breaks the execution cycle in elementary actions. That is, a robot can be activated/turned off while it executes a cycle. Hence, robots are not anymore synchronized.

In both Corda and Suzuki-Yamashita model several problems have been studied under different assumptions on the environment (schedulers, fault-tolerance), robots visibility, accuracy of compasses: circle formation, pattern formation, gathering [6, 7, 8, 9, 10, 11, 12]. The *flocking problem* although largely discussed for real robots ([13, 14] and [15]) was studied from theoretical point of view principally by Prencipe [16, 17]. The authors propose non-uniform algorithms where robots have basically two roles: leader or follower. The leader is unique and all the followers know the leader robot. Obviously, when the leader crashes or disappears or duplicates the flock cannot finish its task. Our approach is different, the leader is not known a priory but it is elected. When the current leader disappears from the system another leader is elected and the network can finish its task. In order to be sound our flocking architecture contains as basic element a leader election module.

The leader election problem has been studied under a broad class of models. Recent works propose solutions in the population protocol model, [18, 19]. The same problem has been also studied in the mobile agents model [20]. These models may seem similar to the robots model however, in these models agents either have a point to point interaction with simultaneous change of the state or assume a specific topology of the network guesting the

agents (eg. rings) or they make additional assumptions like the existence of whiteboards on the nodes visited by agents.

In the robot networks there is no such assumptions since robots evoluate in a Cartesian two dimensional space and they are helped only by the information. In robot networks leader election have been studied in Suzuki-Yamashita model [5]. The authors propose a solution where robots share the same coordinate space. The model we study the leader election is the Corda model and we do not make the assumption of common coordinates. Further in [21] is proposed an algorithm for leader election based on Lyndon words which works if the number of the robots is prime and robots are not disposed in a regular n-gon. In [22] the author prove the leader election impossibility in CORDA model when the number of robots is even.

2 Our contribution

In this paper we propose and prove correct an architecture for a self-organizing and stabilizing flocking system. Contrary to the existing work on this topic our flocking architecture do not relies on the existence of a specific leader a priory known to every node in the network. In our approach robots are uniform, start in an arbitrary configuration and the head of the group is democratically elected via algorithmic tools.

Our architecture includes three modules: a leader election module, a preprocessing module and a motion module. The leader election module returns to each robot its status : leader or follower. The preprocessing module aims to arrange robots in a moving formation. The motion module provides the rules that will make robots change their positions whenever the leader moves. Every modification of robots position will be done maintaining alive the moving formation. For each of these modules we propose deterministic or probabilistic algorithms (in the case when a deterministic solution is impossible). Moreover, we prove their correctness.

3 Model

Most of the notions presented in this section are borrowed from [16, 1]. We consider a system of autonomous mobile robots that work in the CORDA model [1].

Each robot is capable of observing its surrounding, computing a destination based on what it observed, and moving towards the computed destination: hence it performs an (endless) cycle of observing, computing, and moving. Each robot has its own local view of the world. This view includes a local Cartesian coordinate system having an origin, a unit of length, and the directions of two coordinate axes (which we will refer to as the x and y axes), together with their orientations, identified as the positive and negative sides of the axes.

The robots are modeled as units with computational capabilities, which are able to freely move in the plane. They are equipped with sensors that let each robot observe the positions

of the others with respect to their local coordinate system. Each robot is viewed as a point, and can see all the other robots in the flock.

The robots act totally independently and asynchronously from each other, and do not rely on any centralized directives, nor on any common notion of time. Furthermore, they are oblivious, meaning that they do not remember any previous observation nor computations performed in the previous steps. Note that this feature gives to the algorithms designed in this model the nice property of self-stabilization [23]: in fact, every decision taken by a robot cannot depend on what happened in the system previously, and hence cannot be based on corrupted data stored in its local memory. The robots in the flock are anonymous, meaning that they are a priory indis- tinguishable by their appearances, and they do not have any kind of identifiers that can be used during the computation. Moreover, there are no explicit direct means of communication; hence the only way they have to acquire information from their fellows is by observing their positions. They execute the same algorithm, which takes as input the observed positions of the robots, and returns a destination point towards which the executing robot moves.

Summarizing, each robot moves totally independently and asynchronously from the others, not having any bound on the time it needs to perform a move, hence a cycle; therefore, a robot can be seen while it is moving; in addition, they are oblivious, and anonymous.

4 Leader election and flocking Problems

Leader election creates a asymmetry whatever the initial configuration. Robots may be in one of the following states: leader or follower and the leader should be unique in the system.

Definition 1 (leader election) *A system of robots verifies the leader election specification iff the following two properties hold:*

- *Safety: The system is in a legal configuration where there is an unique robot in the state leader and all the other robots are in the state follower.*
- *Liveness: The legal configuration is reached in a finite number of steps.*

Leader election is the building block for a large class of problems. In this paper we focus on the flocking problem. Intuitively, a flock is a group of robots that moves in the plane in order to execute a task while maintaining a specific formation. The most current definition of the flocking implicitly assumes the existence of an unique leader of the group that will lead the group during the task execution. Robots have as input the same pattern representing the flock to be maintained which is described as a set of coordinates in the plane, relative to a point representing the leader.

Obviously, in order to achieve flocking robots need to re-organize their formation whenever the leader change its position. Therefore the definition of flocking has to capture the dynamicity of the flock.

Formally, the flocking problem can be defined as follows:

Definition 2 (flocking) *Let S a system of robots and $\mathcal{P}$ the flocking pattern. S verifies the flocking specification iff the robots satisfy the flocking pattern infinitely often.*

5 Architecture of a flocking system

In the following we define a possible architecture for a fault-tolerant flocking system. The architecture is composed of three modules : the leader election module, the preprocessing module and the flocking module.

- The leader election module is the bases of the architecture. This module accepts as input a set of robots arbitrarily distributed in the plane and elects a leader.
- The preprocessing module prepares the group of robots for the moving formation. All robots but the leader are placed on the smallest enclosing circle. Then all robots on the smallest enclosing circle form a circular moving formation using the leader computed by the leader election module as reference point. One robot in this set will further act as the head of the flock.
- The flocking module receives as input a moving formation which initially has a circular form defined by a reference robot and a head and provides the necessary rules to move this formation in the plane whenever the head changes its position. The objective of the flocking module is to ensure the formation moving while keeping its properties.

6 Leader Election Module

In this section we prove the impossibility of deterministic leader election. Generally, the impossibility results can be circumvent by using randomization. In the following we show that probabilistic leader election is impossible for 2 robots systems. However, the probabilistic leader election is possible for systems of size greater than 3.

6.1 Impossibility results for leader election

In this section we prove the deterministic leader election impossible in Suzuki Yamashita and Corda models. [1]

Theorem 1 *Deterministic leader election is impossible.*

Proof: Lets consider n robots forming a regular n-gon with the local $x - y$ coordinates of each robot such that the y positive axe is directed towards the next robot in clockwise. Assume also the x positive axe is such that the n-gon has no value of x less than 0. Consider all robots have the same unit of length. Without restraining the generality we consider in the following an equilateral triangle. For a deeper comprehension, lets consider Figure 1.

[1] Note that in [22] is proven the impossibility of leader election for n even, while in [21] is shown that leader election can be deterministically solved for n prime and robots not disposed in a n-gon.

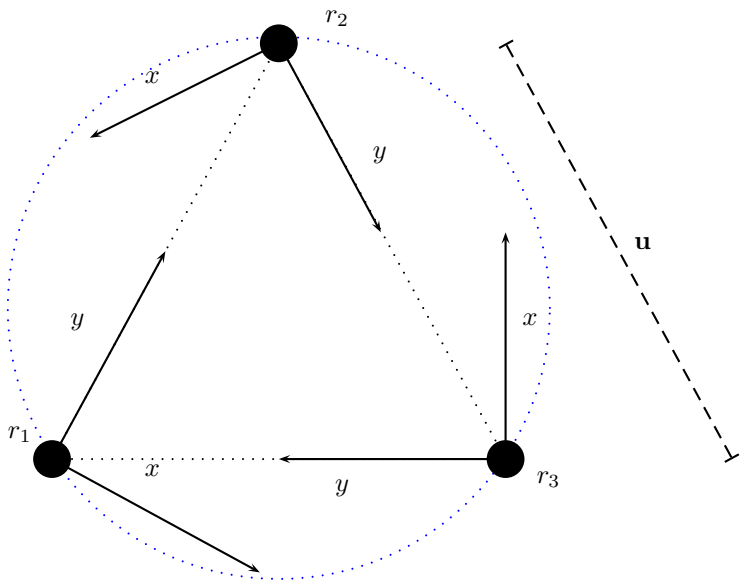

Figure 1: Symmetric Configuration

Each robot can see a robot in (0,0) (itself) and other two robots in $(\frac{u}{2}, \frac{\sqrt{3}}{2}u)$ and in $(u, 0)$. Note that the three robots have the same view.

Assume a configuration such that the leader is the robot in $(u,1)$. Hence, in our example for r_1 the leader is r_2, for r_2 is r_3 and for r_3 is r_1. Hence, each robot sees a different leader. Therefore, the safety property is violated.

Assume an initial configuration where there is no leader. In order to reach a legal leader election configuration robots should move. Assume the algorithm executed by each robot makes them move towards a point (x', y') of their system of coordinates and assume the scheduler chooses all robots to move concurrently. The system reaches a configuration where the n-gon structure is maintained. Moreover, in the new configuration robots have the same view. So, each deterministic movement from a symmetric configuration lead up to symmetric configuration. Hence, the system never converges to a legal configuration. ■

Lemma 1 *There is no probabilistic 2-robots leader election.*

Proof: [sketch] In the following we show for the case of 2 robots even randomization cannot help in solving the problem. Intuitively, assume there exist a probabilistic algorithm. In our model (where robots cannot communicate and they don't have neither memory nor identity), the robots can only elect the leader based on the position of the other robot. Since the two robots neither share the same reference system, nor know the system of the other robot, they can never know when the algorithm conditions are verified for the other robot.

Therefore, the system will contain executions where the two robots may think that the other robot is the leader or executions where the two robots will change infinitely their state(position) in order to become leader. The set of executions verifying the above has probability 1. ■

6.2 Probabilistic leader election

In this section we propose probabilistic solutions for leader election for systems with three or more robots.

6.2.1 Probabilistic leader election with 3 robots

The algorithm idea is to exploit the asymmetry of a triangle. We choose as leader candidate the robot with the smallest angle or the robot different from the other two robots in the case of an isosceles triangle. The randomization is used only to break the symmetry of equilateral triangles. For this particular case we use randomization in order to create an asymmetric triangle on which we apply the method described above.

1) Compute the angle between every two robots.
2) **if** *my_angle* is the smallest
   then become *Leader*.
3) **else if** *my_angle* is not the smallest but the other two are identical
   then become *Leader*.
4) **else if** All the angles are identical
   then move perpendicular to segment linking the other two robots in opposite direction with probability $\frac{1}{3}$

Algorithm 6.1: Leader election algorithm

Lemma 2 *Algorithm 6.1 converges to the leader election specification in a finite number of steps.*

Proof: The proof shows that in finite number of steps equilateral triangles can be broken and transformed in triangles where the election can be done in one step.

1. In the case of scalene or isosceles triangle the algorithm converges in one step.

2. If the triangle is equilateral then the robots apply rule 4 and move with probability $\frac{1}{3}$. In the case when all the three robots move concurrently or all the three robots keep their position then the new configuration is an equilateral triangle with probability $(\frac{1}{3}^3+\frac{2}{3}^3) = \frac{1}{3}$. Otherwise the robots will form a triangle where the election can be done in one step. The probability to break the equilateral triangle after i steps is $\frac{2}{3}(\frac{1}{3})^{i-1}$. In expectation, the number of steps needed to converge is $\sum_{i=1}^{\infty} i\frac{2}{3}(\frac{1}{3})^{i-1} = \frac{3}{2} = 1.5$.

   Once the symmetry broken robots are in the configuration situation discussed at (1).

Overall, the algorithm converges to the leader election specification in a finite number of steps. ■

6.2.2 Probabilistic leader election with more than 3 robots

In the following we propose a leader election algorithm for systems with more than three robots. Note that our leader election works also in symmetrical systems. Intuitively, the leader robot will be the robot which position is the closest to the center of the smallest enclosing circle (SEC). Additionally, we would like the leader to define a second reference together with the center of SEC. Therefore, the leader should not be placed on the center position. If initially a robot is positioned on the center of the smallest enclosing circle than a preprocessing phase is executed. The robot on the center moves to a free position chosen non-deterministically inside the SEC. The leader election algorithm idea is as follows. Robots randomly change their positions until only one of them is the closest to the SEC.

1) Compute the smallest enclosing circle SEC.
2) Compute the distance d_myself to the center of SEC.
3) **if** ($d\_myself < d_k \ \forall$ k$\neq myself$, where 1≤k≤n)
   than { become leader;
   exit; }
4) **if** ($d\_myself \leq d_k \ \forall$ k$\neq myself$, where 1≤k≤n)
   than { move to the center of SEC with probability p of a
   distance $d\_myself \cdot p$)}

Algorithm 6.2: Leader election in systems of size ≥ 3

Definition 3 *A robot is Leader if it is the unique robot closest to the center of the smallest enclosing circle.*

Definition 4 *A robot is a Leader Candidate if it belongs to the set of the robots closest to the center of the smallest enclosing circle.*

Definition 5 (legitimate configuration) *A legitimate configuration is a configuration with a Leader robot.*

Lemma 3 *Algorithm 6.2 is silent.*

Proof: The leader does not change its position and once all robots but the leader are in the set *Placed* no robot can execute its actions so the algorithm is silent. ■

Lemma 4 *Starting in an illegitimate configuration the system converges to a legitimate configuration in a finite number of steps.*

Proof: In an illegitimate configuration the set of Leader Candidates contains more than one robot. These robots are enabled for rule 4 so if chosen by the scheduler they move to the center with a constant probability p.

Assume without restraining the generality only two robots r_i and r_j in Leader Candidate set. We prove that in finite number of steps one of them becomes leader. Initially, the distances to the center of SEC d_{r_i} and d_{r_j} are equal so the only way to have $d_{r_i} \neq d_{r_j}$ is that one of them moves towards the center while the other keeps its position. After s steps the probability that at each step both moved together equals q=$(p^2 + (1-p)^2)^s$. After $s \to \infty$ steps the probability that at each step the robots moved together is:

q= $\lim_{s\to\infty} (p^2 + (1-p)^2)^s$=0

Deriving $p^2 + (1-p)^2$ we can find that the only minimum of the function is in $p = \frac{1}{2}$.

Let X_t be the probabilistic variable that models the configuration of the system at time t. Let $\mathcal{L}$ be the set of terminal configurations (a robot is leader). Let $T_{\mathcal{L}}$ be the time to reach a configuration in $\mathcal{L}$, $T_{\mathcal{L}} = min\{t, X_t \in \mathcal{L}\}$. The convergence time of the algorithm is :

$$\mathbf{E}[T_{\mathcal{L}}] = \sum_{i=1}^{\infty} i[2p(1-p)][p^2 + (1-p)^2]^{i-1} = \frac{2p(1-p)}{[1-(p^2+(1-p)^2)]^2} = \frac{1}{2p(1-p)}$$

If for example the value of p is $\frac{1}{2}$, $\mathbf{E}[T_{\mathcal{L}}] = 2$.

Consider the general case: all the n robots are leader candidates.

$$\mathbf{E}[T_{\mathcal{L}}] = \sum_{i=1}^{\infty} i[np(1-p)^{n-1}][1-np(1-p)^{n-1}]^{i-1} = \frac{np(1-p)^{n-1}}{[1-(1-np(1-p)^{n-1})]^2} = \frac{1}{np(1-p)^{n-1}}$$

Let's compute the value of p that minimize $\mathbf{E}$.

$$\frac{d}{dp}(\frac{1}{np(1-p)^{n-1}}) = \frac{n(1-p)^{n-1} - np(n-1)((1-p)^{n-2}}{(np(1-p)^{n-1})^2} = \frac{p(-n^2+2n)+n}{(np)^2(1-p)^n}$$

.

But we have a minimum if the derivate is equal to 0:

$$\frac{p(-n^2)+n}{(np)^2(1-p)^n} = 0 \Rightarrow p = \frac{n}{n^2} = \frac{1}{n}$$

Finally we have a minimum if $p = \frac{1}{n}$ and for this value $\mathbf{E}[T_{\mathcal{L}}] = \frac{1}{(1-\frac{1}{n})^{n-1}}$.

■

7 Preprocessing Module : Setting a Moving Formation

In this section we gradually set the motion pattern used further in the flocking algorithm. We build upon the leader election algorithms developed in Section 6.2. The construction takes two phases. First, all robots but the leader will be placed on the smallest enclosing circle. Then, the robots on the circle will be placed in their final positions for motion.

7.1 Phase 1 : Placement on the smallest enclosing circle

In this section we propose an algorithm for placing robots on the smallest enclosing circle. This algorithm uses as building block the algorithm proposed in the previous section for the leader election. Once this algorithm stabilizes all robots but the leader are placed on the smallest enclosing circle. Note that, the leader does not change during this phase.

The algorithm works "in waves". First, the robots closest to the bounds of the smallest enclosing circle are placed. Than recursively all the other robots but the leader are placed. The robots that should occupy a position that is already occupied by another robot will be placed on a free position between the robot that occupied their position and the next one on the smallest enclosing circle. We assume the robots agree on the same direction of the Ox axe given by the center of SEC and the leader and the same direction of Oy axe. Our algorithm is collision free and works in the CORDA model. Note that in [24] the authors propose similar deterministic solutions for Suzuki-Yamashita model. Interestingly, our algorithm has the same time complexity as the solution proposed in [24].

Definition 6 (FreeToMove) *Let $FreeToMove$ be the set of robots without robots between themselves and the SEC (including the border) along the radius passing through them, and that do not belong to the border of the SEC.*

Definition 7 (Placed) *Let $Placed$ be the set of robots belonging to the border of the SEC.*

Definition 8 *A legitimate configuration for Algorithm 7.1 is a configuration where all robots but the leader are $Placed$.*

$\forall r_i$ compute the value of the radius passing through r_i. Let rad_{r_i} be the value of the angle between my radius ($rad_{myself} = 0$) and the radius of robot r_i, in clockwise direction
$\forall r_i$ compute the value of $dist_{r_i}$, distance of the robot r_i to the border of smallest enclosing circle (SEC)

Predicates:
$Leader(myself) \equiv \forall r_i$ with $i \neq myself$, $dist_i < dist_{myself}$

Functions:
$OccupiedPosition(rad_{myself})$: returns $r_i, i \neq myself, dist_{r_i} = 0$ *and* $rad_{r_i} = rad_{myself}$ otherwise $\perp$
$NextToMove$: returns the set of closest robots r to the SEC with $dist_r \neq 0$

1) **if** ($\neg Leader(myself) \wedge myself \in FreeToMove$)
than { move to SEC with distance $dist_{myself}$}
2) **if** ($\neg Leader(myself) \wedge (myself \in NextToMove) \wedge (FreeToMove = \emptyset) \wedge (OccupiedPosition(rad_{myself}) \neq \perp)$)
than { Move to the middle point of the arc between robot OccupiedPosition(rad_{myself}) and robot r_j belonging to the SEC such that rad_j is minimum.}

Algorithm 7.1: Positioning Algorithm executed by robot my_self

Note that the algorithm does not change the leader position neither the position of *Placed* nodes. Moreover, there is no node behind the leader and sharing the same radius as the leader. Otherwise this node will be the closest to the center of the SEC hence the real leader.

Lemma 5 *Algorithm 7.1 is silent.*

Proof: The leader does not change its position and once all robots but the leader are in the set *Placed* no robot can execute its actions so the algorithm is silent. ■

Lemma 6 *If two robots r_i and r_j belong to the set $FreeToMove$, than their final position will be different.*

Proof: If there are no robots between r_i and the SEC along $radius_i$ and there are no robots between r_j and the SEC along $radius_j$ than $radius_j$ and $radius_i$ must be different. To different radius correspond different positions on the SEC so r_i and r_j, thanks to Rule 1, will be placed on different positions. ■

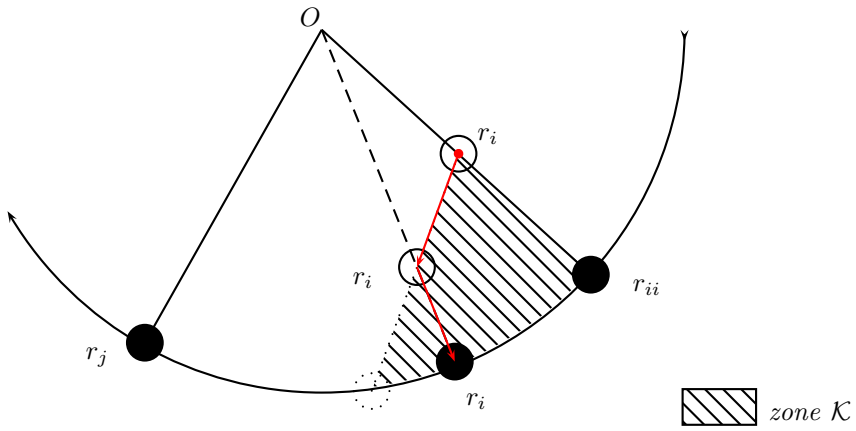

Figure 2: Proof of Lemma 8

Lemma 7 *A robot always moves towards a free position on the SEC.*

Proof: Rule 1 moves robots in the set $FreeToMove$. By definition these robots move only on free positions.

A robot not in the set $FreeToMove$ is allowed to move only when the set $FreeToMove$ is empty and the robot is in the set $NextToMove$. Let r_i be such robot. r_i is enabled for Rule 2 and moves towards the middle point of the arc between the robot r_{ii} and the next robot (r_j) on the border of SEC (see Figure 2). Note that the arc between the two robots is free (otherwise r_j would not be the next after r_{ii}). Moreover, the sector between r_j and r_i is free. Assume a robot exists in that area. Following Rule 1 it must go towards SEC before r_i moves and hence becoming the next robot of the SEC after r_{ii}. Since r_i can move freely in this sector, the middle point is free too and r_i can move directly to it. If instead it cannot reach the middle point in one move (due to scheduler interference), at the next one it will still verify the conditions of Rule 1, so it will move to the SEC following this same rule.

■

Lemma 8 *Algorithm 7.1 is collisions free (two robots never move towards the same free position).*

Proof: Firstly if two robots r_i and r_j start at different distances from the SEC, only the one closer to SEC can move, the other one must wait until the former reaches the border. If the two robots r_i and r_j have the same distance from SEC, than they may move at the same time. If they are enabled for Rule 1 then they never collide since they will reach different positions on the SEC moving along their respective radius which are different (Lemma 6).

If both r_i and r_j are enabled for Rule 2, then both robots must move towards the middle point between the robot on SEC belonging to its radius and the next robot on the SEC in clockwise direction. Thanks to Lemma 7 the sector between these robots is free and a robot can move only inside the zone $\mathcal{K}$ (see Figure 2). Let r_{p_i} and r_{p_j} be the robots at the intersection between the radius of r_i receptively r_j and SEC. Let r_{pnext_i} and r_{pnext_j} the next robots on SEC for r_{p_i} and r_{p_j}. Since r_i and r_j do not belong to the same radius so $r_{p_i} \neq r_{p_j}$ and $r_{Pnext_i} \neq r_{Pnext_j}$. Hence the two sectors where r_i and r_j can move have no intersection but they can be consecutive if $r_{pnext_i} = r_{p_j}$ or $r_{pnext_j} = r_{p_i}$. From the above r_i never reaches r_{pnext_i}.

Overall r_i and r_j will never reach the same final position nor meet on the way to their respective final positions. ■

Lemma 9 *Algorithm 7.1 converges in a finite number of steps, $O(n)$, to a legitimate configuration.*

Proof: Rule 1 allows only robots in $FreeToMove$ set to move to the SEC. Thanks to Rule 2 all robots that don't belong to this set wait until it is empty. Once this set is empty, the robots, excepted the leader, that are not on the SEC can execute Rule 2. This rule can be executed only by the set of robots i with $min(dist_i)$ where $dist_i \neq 0$. Now these robots can move towards the middle point of the arc of SEC between the robot on SEC belonging to its radius and the next robot on the SEC in clockwise direction. Once these robots arrive on the border of SEC and their corresponding $dist$ equals 0, other robots can satisfy Rule 2 and move to the SEC. This process is iterated until all robots but the Leader are on the border of SEC. Thanks to Lemma 8 robots do not collide and no robot obstructs the trajectory of some other robots.

In the following we show in the worst case the algorithm converges in $O(n)$ steps. Assume all robots share the same radius. In order to place a robot on SEC, all robots that precede it should be placed. So, robot number i should wait for $i-1$ robots. Hence, the complexity of the algorithm is $O(n)$. ■

7.2 Phase 2: Setting the flocking configuration

In this section we propose an algorithm that starting from the final configuration of Algorithm 7.1 reaches a flocking pattern or moving formation having the singularity property detailed later.

Initially, we place robots in a circular moving formation then in the final moving formation. The circular moving formation has the following shape: r_0 is inside SEC (the one computed by Algorithm 7.1) and all the other robots are disposed on its border. These robots are placed as follows: a robot r_1 is in the position $SEC \cap [O, r_0)$ and the others, uniformly disposed on the semi-circle that does not contain r_1 and which ends in the points given by the intersection of SEC with the perpendicular on $[O, r_0)$ that passes through O ($SEC \cap (\perp[O, r_0)$ *on* O)). In the following this configuration will be referred as *circular moving formation* (see Figure 3 for a seven robots example).

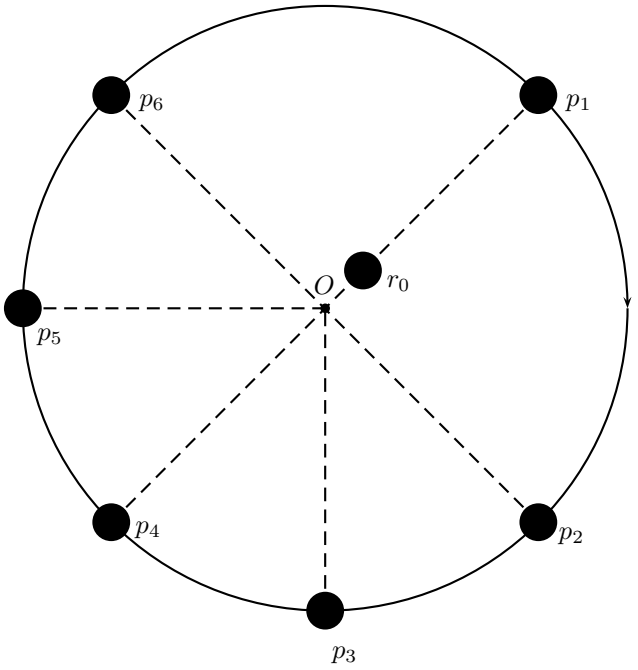

Figure 3: Circular moving formation

In order to construct the circular moving formation we use the concept of oriented configuration [21]:

Definition 9 (oriented circular configuration) *A configuration is called circular oriented if the following conditions hold:*

1. *All robots are at distinct positions on the same circle SEC, except only one of them, called r_0, located inside SEC ;*
2. *r_0 is not located at the center of SEC;*

Note Algorithm 7.1 verifies point 1 of the definition above and is collisions free contrary to the solution proposed in [21]. Note also the leader election algorithm chooses a leader such that it is the closest to the center of SEC without reaching this center. If the leader is initially in the center, we recall that a preprocessing is performed in order to take care of this particular case. The leader election algorithm is executed only after the end of the preprocessing phase.

We now describe Algorithm 7.2. The algorithm make use of the following function: FinalPositions(SEC, p_1) which returns, when invoked by a robot, the set of positions in the circular moving formation with respect to SEC and the point p_1. p_1 is the intersection between the segment $[O, r_0)$ and the circle SEC. The order of positions and robots is given clockwise starting with position p_1. Started in an oriented configuration Algorithm 7.2 eventually converges to a configuration where robots are disposed on SEC following the restrictions imposed by the $FinalPositions$ function.

Functions:
$get\_number(myself)$ returns the number of robots between $myself$ and position p_1 (including robot myself) clockwise
$get\_position(myself)$ returns the position $get\_number(myself)$ in $FinalPositions(SEC, p_1)$
$FreeToMove(myself)$ returns true if there are no robots between $myself$ and $get\_position(myself)$

Motion Rule:
if FreeToMove(myself) **than**
move to $get\_position(myself)$

Algorithm 7.2: Setting the moving formation executed by robot $myself$

The idea of the algorithm is as follows. Robots started in an oriented configuration reach their final positions. If a robot is blocked by some other robot than it waits until this robot is placed in its final position. In the following we prove no robot is blocked infinitely.

Lemma 10 *In a system with n robots Algorithm 7.2 started in an oriented configuration converges in finite number of steps, $O(n)$, to a configuration where all robots reach their final positions computed via FinalPositions function.*

Proof: Let $p_1, \ldots, p_n$ be the robots' final positions returned by $FinalPositions(SEC, p_1)$ where $p_1 = SEC \cap [O, r_0)$ and let $r_1, \ldots, r_n$ be the set of robots to be placed. Assume the worst initial configuration: no robot it placed in its final position Let denote L the segment defined by the robots not placed in their final positions. The initial length of L is n. The robot r_1 (the first robot in L) can freely move to its final position p_1. Once this robot is placed the length of segment L becomes $n-1$. One of the new ends of L can be placed to its final position. Assume the contrary. All robots are blocked. So, there is an waiting chain such that $r_2 \rightarrow r_3 \rightarrow \ldots \rightarrow r_n$ or $r_n \rightarrow \ldots r_2$. Since the chain is finite and not cyclic (due to the total order of these positions) one of the ends of the chain can move (r_2 or r_n). After this robot moves and the length of the segment L decreases. Eventually, all robots in L finish in their final positions. In the worst case, the algorithm converges in $O(n)$ steps. ■

We define formally the moving formation as follows:

Definition 10 (moving formation) *A set of $n > 4$ robots, $r_0, \ldots r_n$, is a moving formation if:*

- *r_1 and r_0 define the Oy axe of the system such that: the y coordinate of r_0 equals 0 and the positive values are in the r_1 direction;*
- *the axe Ox is perpendicular to Oy in r_0 and has positive values at the right of Oy;*
- *all the other robots are such that:*

  1. $\forall\ r_i \neq r_l$ *and* $r_i \neq r_0 \Rightarrow y_{r_i} < 0$
  2. $\forall r_i, r_j,\ x_{r_i} \neq x_{r_j}$;
  3. $\forall\ r_i,\ \exists\ r_j$ *such that* $x_{r_i} = -x_{r_j}$
  4. *if* $|x_{r_i}| > |x_{r_j}|$ *than* $|y_{r_i}| < |y_{r_j}|$
  5. *there exists an unique robot with* $x = 0$ *and* $y < 0$

In the following we prove the unicity(singularity) property of the moving formation defined above. More precisely, we show that there is only one formation that satisfies Definition 10 when $n > 4$. Note that for the case $n \leq 4$ the formation defined by Definition 10 is not unique. In the sequel we consider systems with more than 4 robots. For the case $n \leq 4$ simple adhoc algorithms can be designed on top of the algorithm proposed in Section 6.2.1.

Lemma 11 *In a set of $n > 4$ robots that verify Definition 10 does not exist a situation with two different r_1 and a common r_0.*

Proof: Assume there exists another r_1' that plays the same role as r_1. In the following r_1 and r_1' will be called leaders (see Figure 4).

Intuitively the proof is as follows. Let α be the smallest angle $\widehat{r_1 r_0 r_1'}$:

- *if* $\alpha \leq 90^o$ then the two r_1 and r_1' have an $y \geq 0$ with respect to the other robot view, this contradictions item (1) of Definition 10.
- *if* $\alpha = 180^o$, then the two leaders face each other and they have a common x axis, but the y direction is inverted so the space is divided in two parts by $x \equiv x'$. In one part $y > 0$ and $y' < 0$ and, in the other one, $y < 0$ and $y' > 0$. So, if it exists a robot r in the first part there will be two robots (r and r_1') with $y > 0$, otherwise, if r is in the second part there will be two robots with $y' > 0$, (r and r_1). But this contradicts item (1) of Definition 10.
- *if* $90^o < \alpha < 180^o$. Firstly, note there exists only a sector $\mathcal{L}$ of the space where a robot can exist ($y < 0 \cap y' < 0$). According to item 3, Definition 10 there must exist a robot r_p such that $x_{r_p} = -x_{r_1'}$ and such that $x'_{r_p} = -x'_{r_1}$. Assume this robot in $\mathcal{L}$.

If there exists another robot $r_q \neq r_p \in \mathcal{L}$ than there must exist a robot r_{q1} such that $x_{r_{q1}} = -x_{r_q}$ and such that $x'_{r'_{q1}} = -x'_{r_q}$. But this point has values of y and y' greater than 0, $(-x_{r_q}, -y_{r_q})$ and $(-x'_{r_q}, -y'_{r_q})$ in the other system, which contradicts point (1) of Definition 10.

Geometrically, the proof is as follows.

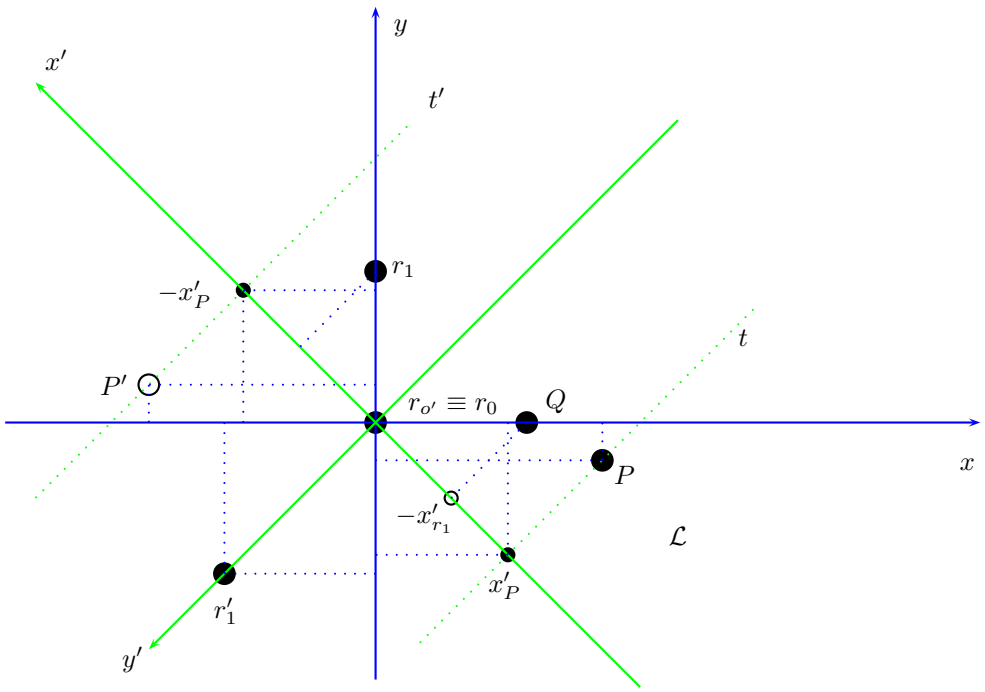

Figure 4: Unicity of the moving formation, Lemma 11

We geometrically prove there is no moving formation with two distinct r_1 and a common r_0 in a system with five or more robots. In particular, even if it exists two distinct r'_1, r_1 and a common r_0, it is possible to have a four robots moving formation however any other robot in $\mathcal{L}$, the only zone that agrees with item (1), will be in contradiction with item (3). As shown in Figure 4 it is possible to find a robot $Q \in \mathcal{L}$ such that $|x_{r'_1}| = |x_Q|$ and $|x'_{r_1}| = |x'_Q|$. In the following we prove that any other point $P \in \mathcal{L}$, according with item (3), will implicate another point P' such that $y_{P'} = -y_P$ and $y'_{P'} = -y'_P$, but this new point is in contradiction with item (1).

We first give the coordinates of the P' in the $x-y$ system. Note that for $(x'-y')$ the computation is symmetric. We will prove further that the two robots P and P' have y values with opposite signs, so one of the two robots will have the same sign as r_1 which contradicts item (1).

Calculation of y' and x' in the $x-y$ coordinates. y' : $y = m'x$ where $m' = \frac{y_{l'}}{x_{l'}}$ so $y = \frac{y_{l'}}{x_{l'}}x$ x' : $y = -\frac{1}{m'}x$ so $y = -\frac{x_{l'}}{y_{l'}}x$ Line t parallel to y' passing through P is t : $y - y_P = m'(x - x_P)$ than $y = \frac{y_{l'}}{x_{l'}}x - \frac{y_{l'}}{x_{l'}}x_P + y_P$ The coordinates of x'_P are:

$$\begin{cases} y = \frac{y_{l'}}{x_{l'}}x - (\frac{y_{l'}}{x_{l'}}x_P - y_P) \\ y = -\frac{x_{l'}}{y_{l'}}x \end{cases}$$

$$\frac{x_{l'}^2+y_{l'}^2}{x_{l'}y_{l'}}x = \frac{y_{l'}}{x_{l'}}x_P - y_P \;\rightarrow\; x = (\frac{y_{l'}}{x_{l'}}x_P - y_P)(\frac{x_{l'}y_{l'}}{x_{l'}^2+y_{l'}^2})$$

$$\begin{cases} x = \frac{y_{l'}x_P - x_{l'}y_P}{x_{l'}^2+y_{l'}^2}y_{l'} \\ y = -\frac{y_{l'}x_P - x_{l'}y_P}{x_{l'}^2+y_{l'}^2}x_{l'} \end{cases}$$

Building the triangle rectangle with $(-x'_P, x'_P)$ as hypotenuse and cathetus parallel to $x-y$ axis, it is easy to notice that $x_{-x'_P} = -x_{x'_P}$ and $y_{-x'_P} = -y_{x'_P}$. Let denote $\frac{y_{l'}x_P - x_{l'}y_P}{x_{l'}^2+y_{l'}^2} = A$ hence $-x'_P = (-Ay_{l'}, Ax_{l'})$.

We proceed calculating t', t' : $y = \frac{y_{l'}}{x_{l'}}x + \frac{y_{l'}}{x_{l'}}Ay_{l'} + Ax_{l'} = \frac{y_{l'}}{x_{l'}}x + A\frac{y_{l'}^2+x_{l'}^2}{x_{l'}}$. So, t' : $y = \frac{y_{l'}}{x_{l'}}x + \frac{y_{l'}x_P - x_{l'}y_P}{x_{l'}}$. Finally, the P' coordinates are :

$$\begin{cases} y = \frac{y_{l'}}{x_{l'}}x + \frac{y_{l'}x_P - x_{l'}y_P}{x_{l'}} \\ x = -x_P \end{cases}$$

$y_{P'} = \frac{1}{x_{l'}}(-y_{l'}x_P - x_{l'}y_P + y_{l'}x_P) = -y_P$

If $y_P < 0$ than $y_{P'} > 0$ and this contradicts item (1). ■

Lemma 12 *In a set of $n > 4$ robots that verify Definition 10 does not exist a situation with two different r_1 and two different r_0.*

Proof: Assume the contrary and let r'_0 and r'_1 the siblings of r_0 and r_1. Consider the following three situations:

- r_0, r_1 and r'_0, r'_1 share the same line: this situation is in contradiction with item (1) Definition 10.
- the couple r_0, r_1 is such that r_0 is at the right hand and r_1 is at the left of y'. In this case r'_0 and r'_1 are both at the left or at the right of y. Now, according to item (3), there must exist a symmetric couple of robots in the other side of y. If this couple exist, it will be on a side (left or right) of y'. But again, according to item (3), there

must exist a symmetric couple of robots in the other side of y'. So if this new couple of robots exists, consequently another couple must exist, an so on. Finally item (3) will never be respected.

- the two couples of robots are both at the right or at the left of the y axe of the other couple (see Figure 5). In this case, if r_0, r_1 and r'_0, r'_1 exist, there must exist another couple of robots (Q and P where $x_P = -x_{r'_1}$ and $x'_P = -x'_{r_1}$) that satisfy the item (3). But we will geometrically prove that this couple of robots is always such that even if $|x_o| < |x_P|$ anyway $|y_o| < |y_P|$ and this contradicts item (4) Definition 10

Geometrically, the proof goes as follows:

We have two couples of robots r_1, r_0 and r'_1, r'_0. Our objective is to calculate, in the $x-y$ coordinates, the value of P, the point such that $|x_P| = |x_{r'_1}|$ and such that $|x'_P| = |x'_{r_1}|$, to demonstrate that the y_P value is always greater than the $y_{r'_o}$ value. This contradicts the definition of the moving formation (item (4)). Note that the computation for x', y' system of coordinates is totally symmetric. To simplify the notation we call $r'_1 = a$, $r'_0 = b$, $r_1 = c$ and $r_0 = d$.

First we compute the value of the inclination of y' axis in the $x-y$ system of coordinates. $m_{y'} = \frac{y_a - y_b}{x_a - x_b}$. Then the equation of the x' axe: $y = -\frac{x_a - x_b}{y_a - y_b}(x - x_b) + y_b$ and the projection of r_1 (c) on x': $\begin{cases} y = \frac{y_a - y_b}{x_a - x_b}(x - 0) + y_c \\ y = -\frac{x_a - x_b}{y_a - y_b}(x - x_b) + y_b \end{cases}$

the intersection equals:
$-\frac{x_a - x_b}{y_a - y_b}(x - x_b) + y_b = \frac{y_a - y_b}{x_a - x_b}(x - 0) + y_c$
$x(-\frac{(y_a - y_b)^2 (x_a - x_b)^2}{(x_a - x_b)(y_a - Y_b)}) = \frac{x_a - x_b}{y_a - y_b} x_b + y_b + y_c$
If
$Q = (-\frac{(y_a - y_b)^2 (x_a - x_b)^2}{(x_a - x_b)(y_a - Y_b)})$
and
$R = \frac{x_a - x_b}{y_a - y_b} x_b + y_b$

than x'_c on x is

$$\begin{cases} x = \frac{R}{Q} \\ y = \frac{(y_a - y_b)R}{(x_a - x_b)Q} + y_c \end{cases}$$

Building the triangle rectangle with $(-x'_c, x'_c)$ as hypotenuse and cathetus parallel to $x - y$ axis, it is easy to notice that $x_{-x'_c} = x_b + (x_b - x'_c)$ and $y_{-x'_c} = y_b + (y_b - y'_c)$

now we can express the position of $-x'_c$:

$$\begin{cases} x = 2x_b - \frac{R}{Q} \\ y = 2y_b - (\frac{(y_a - y_b)R}{(x_a - x_b)Q} + y_c) \end{cases} .$$

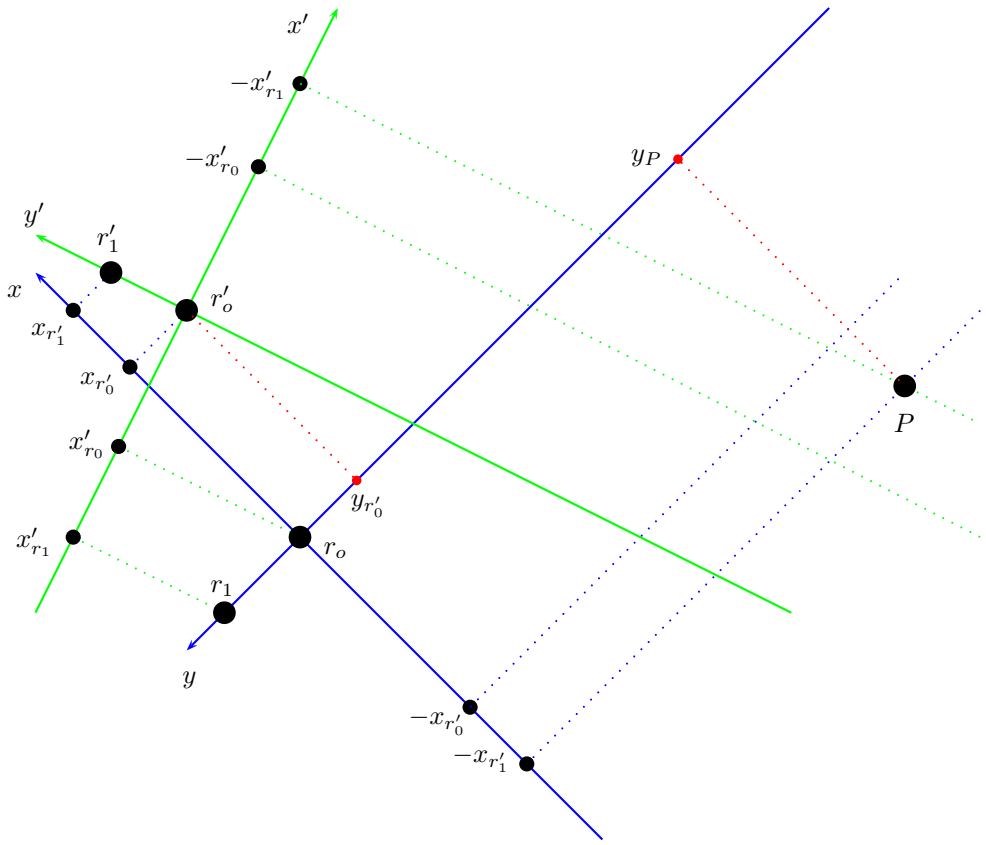

Figure 5: Unicity of the moving formation Lemma 12

Finally the point P is the intersection point between $\begin{cases} y = \frac{y_a - y_b}{x_a - x_b}(x - x_{-x'_c}) + y_{-x'_c} \\ x = -x_a \end{cases}$

the y value of the point P is:
$y = \frac{y_a - y_b}{x_a - x_b}(-x_a - 2x_b + \frac{R}{Q}) + 2y_b - (\frac{(y_a - y_b)R}{(x_a - x_b)Q} + y_c) =$
$= -\frac{(y_a - y_b)(x_a + 2x_b)}{x_a - x_b} + 2y_b - y_c$

Lets now study the sign of $k = -\frac{(y_a - y_b)(x_a + 2x_b)}{x_a - x_b} - y_c$

By the initial thesis, x_a and x_b have both the same sign and x_b is smaller in modulus. So, if x_a and x_b are both negative, $(x_a + 2x_b)$ and $(x_a - x_b)$ are both negative, otherwise they are both positive. Finally their ratio is always positive.
Moreover, if $|x_b| < |x_a|$, then by item (4) $|y_b| > |y_a|$ but being both negative by item (4), $y_b < y_a$ and so $(y_a - y_b) > 0$. Finally $y_c > 0$, because of the definition of r_1
Now we know that k is always positive. And so, $y_P = -k + 2y_b \Rightarrow y_P > 2y_b \Rightarrow y_P > y_b$
Finally we have demonstrated that $y_P > y_b$, and this is in contradiction with item (4). ■

Theorem 2 *The moving formation defined by Definition 10 is unique when $n > 4$.*

Proof: The proof is the direct consequence of Lemmas 11 and 12. ■

Corollary 1 *Algorithm 7.2 started in an oriented configuration eventually place n robots in a circular moving formation if FindPositions returns a set of positions verifying Definition 10.*

8 Flocking Module

In this section we propose a flocking algorithm. The flock of robots verifies the moving formation defined in Definition 10 and follow the head robot (the robot referred as robot r_1) whenever this head changes its position. In the following the robot r_0 of the moving formation will be referred as *reference robot* and the robot r_1 will be referred as the *leader*. The only constraint imposed to the system is: the leader cannot move quicker than the slowest robot in the set. The algorithm idea is as follows. When the head of the group moves, it is followed within a distance δ (a parameter of the algorithm) by the reference robot. Then the closer robots to the reference move within a distance ϵ to the reference and so on till all the robots in the group move. Note the algorithm has three parameters: the speed of the leader, the distance between the leader and the reference and the distance ϵ between the successive rows of robots. The moving formation can be seen as a virtual tree where levels are linked to each other via virtual springs (Figure 8).

Lemma 13 *Algorithm 8.1 preserves the moving formation (Definition 10).*

Proof: In the following we prove the moving formation is preserved after each step of Algorithm 8.1.

1. the initial y axe is always preserved because r_0 and r_1, the robots that define the axe, can move only along it;
2. By Rule 2, r_0 never surpasses r_1, so the y axe never changes its direction.
3. By Rule 3, no robot can surpass r_0. That is, the most external robots never surpass the x axe and so r_0, that is in (0,0). By Rule 4, no robot can surpass its closest external neighbour. Therefore item 1 of the moving formation, Definition 10 is respected.

Input: $r_0, r_1 \ldots r_n$ a moving formation

Functions:
$TheMostExterior(r_{myself})$ returns true if $|x_{r_{myself}}| \geq |x_{r_i}|\ \forall r_i$
$YClosestExterior(r_{myself})$ returns the y coordinate of r_{ext}, the robot such that $(|x_{r_{ext}}| - |x_{r_{myself}}|)$ is minimum and positive

1. **if** $(r_{myself} == r_1)$: { move ahead at a speed $< \ v_{max}$ }, v_{max} is a parameter of the algorithm

2. **if** $(r_{myself} == r_0)$: { follow the leader within a distance δ }

3. **if** $(r_{myself} \neq r_0, r_1 \ \& \ TheMostExterior(r_{myself}))$: { move ahead following $y = y_{r_j}$ towards the point $(x_{r_j}, \text{-}\epsilon)$; }

4. $(r_{myself} \neq r_0, r_1 \ \& \neg TheMostExterior(r_{myself}))$: { move ahead following $y = y_{r_j}$ towards the point $(x_{r_j}, YClosestExterior(r_{myself}) - \epsilon)$; }

Algorithm 8.1: Flocking executed by robot r_{myself}

4. No robot change its x value. Therefore, items 2, 3 and 5 of the moving formation definition are respected.

5. Rule 4 ensures that no robot can surpasses its closest external neighbour so, if x_i is external to x_j, in the initial flocking formation $|y_i| < |y_j|$, then after the execution of the flocking algorithm, $|y_i|$ will always be smaller than $|y_j|$ and the item 4 of the moving formation is respected.

■

9 Conclusions and Open Problems

In this paper we proposed and proved correct an architecture for building a self-organizing and stabilizing flocking architecture. Contrary to the existing work on this topic our flocking architecture do not relies on the existence of a specific leader a priory known to all nodes in the network. In our approach robots are uniform and the head of the group is democratically elected via algorithmic tools.

Our architecture includes three modules: a leader election module, a preprocessing module and a motion module. For each of these modules we propose deterministic or probabilistic

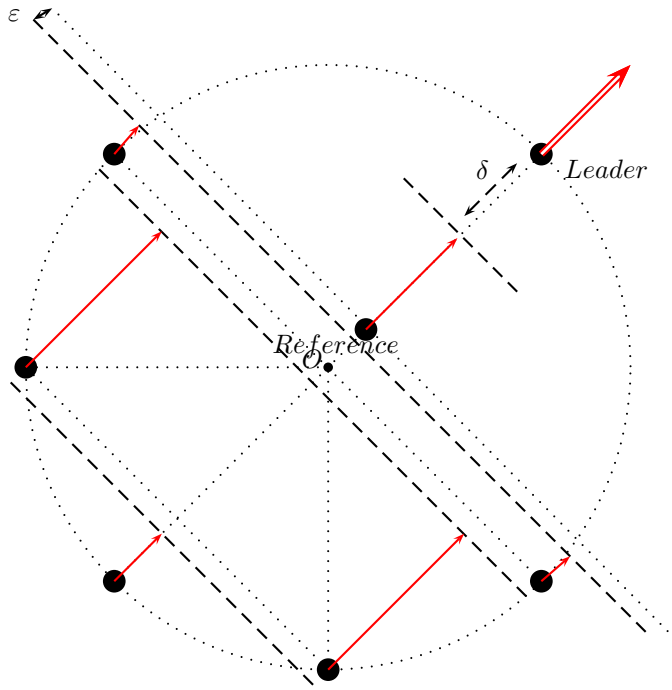

Figure 6: Animation for Algorithm 8.1

algorithms (in the case when deterministic solutions are impossible). Moreover, we prove their correctness and compute their complexity.

This work can be seen as a preliminary study for the design of a general fault-tolerant flocking architecture where the group of robots verify a generic pattern and follow the head whatever its direction. Additionally, we currently investigate probabilistic algorithms that improve the leader election part of our architecture.

10 Acknowledgments

The authors would like to thank Xavier Defago for the hints and discussions related to the current work and the suggested open problems included in Section 9.

References

[1] G. Prencipe. Corda: Distributed coordination of a set of autonomous mobile robots. *Proc. ERSADS, pages 185–190, May 2001.*, 2001.

[2] P. Flocchini, G. Prencipe, N. Santoro, and P. Widmayer. Distributed coordination of a set of autonomous mobile robots. *IVS, pages 480-485, 2000.*, 2000.

[3] I. Suzuki and M. Yamashita. Distributed anonymous mobile robots—formation and agreement problems. *Proceedings of the 3rd International Colloquium on Structural Information and Communication Complexity (SIROCCO '96), Siena, Italy, June 1996.*, 1996.

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

[5] Masafumi Suzuki, Ichiro ; Yamashita. A theory of distributed anonymous mobile robots formation and agreement problems. Technical report, WISCONSIN UNIV-MILWAUKEE DEPT OF ELECTRICAL ENGINEERING AND COMPUTER SCIENCE, 6 1994.

[6] P. Flocchini, G. Prencipe, N. Santoro, and P. Widmayer. Gathering of asynchronous mobile robots with limited visibility. *Theoretical Computer Science*, 337:147–168, 2005.

[7] S. Souissi, X. Défago, and M. Yamashita. Eventually consistent compasses for robust gathering of asynchronous mobile robots with limited visibility. Research Report IS-RR-2005-010, JAIST, Ishikawa, Japan, July 2005.

[8] 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, October 1999.

[9] R. Cohen and D. Peleg. Convergence of autonomous mobile robots with inaccurate sensors and movements. In B. Durand and W. Thomas, editors, *23rd Annual Symposium on Theoretical Aspects of Computer Science (STACS'06)*, volume 3884 of *LNCS*, pages 549–560, Marseille, France, February 2006. Springer.

[10] N. Agmon and D. Peleg. Fault-tolerant gathering algorithms for autonomous mobile robots. In *Proc. 15th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA 2004)*, pages 1070–1078, New Orleans, LA, USA, January 2004.

[11] Xavier Défago, Maria Gradinariu, Stéphane Messika, and Philippe Raipin Parvédy. Fault-tolerant and self-stabilizing mobile robots gathering. In *DISC*, pages 46–60, 2006.

[12] Masao Kasuya, Nobuhiro Ito, Nobuhiro Inuzuka, and Koichi Wada. A pattern formation algorithm for a set of autonomous distributed robots with agreement on orientation along one axis. *Systems and Computers in Japan*, 37(10):89–100, 2006.

[13] A. Qadi Jiangyang Huang, S.M. Farritor and S. Goddard. Localization and follow-the-leader control of a heterogeneous group of mobile robots. *Mechatronics, IEEE/ASME Transactions on*, 11:205–215, 2006.

[14] P. Renaud, E. Cervera, and P. Martiner. Towards a reliable vision-based mobile robot formation control. *Mechatronics, IEEE/ASME Transactions on*, 4:3176– 3181, 2004.

[15] Magnus Lindhe. *A flocking and obstacle avoidance algorithm for mobile robots*. PhD thesis, KTH Stockholm, 2004.

[16] Vincenzo Gervasi and Giuseppe Prencipe. Flocking by a set of autonomous mobile robots. Technical Report TR-01-24, Universitat di Pisa, 2001.

[17] V. Gervasi and G. Prencipe. Coordination without communication: The case of the flocking problem. *Discrete Applied Mathemathics, 2003*, 2003.

[18] D. Angluin, J. Aspnes, M. Fischer, and H. Jiang. Self-stabilizing population protocols. *In Ninth International Conference on Principles of Distributed Systems, pages 79– 90, December 2005.*, 2005.

[19] Michael Fischer and Hong Jiang. Self-stabilizing leader election in networks of finite-state anonymous agents. In *OPODIS*, pages 395–409, 2006.

[20] L. Barri, e Flocchini, P. Fraigniaud, and N. Santoro. Electing a leader among anonymous mobile agents in anonymous networks with sense-of-direction. Technical Report 1310, Technical Report LRI, Laboratoire de recherche en Informatique, Université Paris-Sud, France, April 2002., 2002.

[21] Yoann Dieudonne and Franck Petit. Circle formation of weak robots and lyndon words. *Inf. Process. Lett.*, 101(4):156162, 2007.

[22] Giuseppe Prencipe. Achievable patterns by an even number of autonomous mobile robots. Technical Report TR-00-11 Universitat di Pisa, 17 2000.

[23] Shlomi dolev. *Self-stabilization*. MIT Press, 2000.

[24] Samia Suissi. *Fault resilient cooperation of Autonomous Mobile robots with unreliable compass sensors*. PhD thesis, JAIST, Japon, 2007.

Unité de recherche INRIA Rocquencourt
Domaine de Voluceau - Rocquencourt - BP 105 - 78153 Le Chesnay Cedex (France)

Unité de recherche INRIA Futurs : Parc Club Orsay Université - ZAC des Vignes
4, rue Jacques Monod - 91893 ORSAY Cedex (France)
Unité de recherche INRIA Lorraine : LORIA, Technopôle de Nancy-Brabois - Campus scientifique
615, rue du Jardin Botanique - BP 101 - 54602 Villers-lès-Nancy Cedex (France)
Unité de recherche INRIA Rennes : IRISA, Campus universitaire de Beaulieu - 35042 Rennes Cedex (France)
Unité de recherche INRIA Rhône-Alpes : 655, avenue de l'Europe - 38334 Montbonnot Saint-Ismier (France)
Unité de recherche INRIA Sophia Antipolis : 2004, route des Lucioles - BP 93 - 06902 Sophia Antipolis Cedex (France)

Éditeur
INRIA - Domaine de Voluceau - Rocquencourt, BP 105 - 78153 Le Chesnay Cedex (France)
http://www.inria.fr
ISSN 0249-6399