Self-Deployment of Mobile Sensors on a Ring

Paola Flocchini * Giuseppe Prencipe† Nicola Santoro‡

Abstract

Unlike their static counterpart, mobile sensors can *self-deploy* in a purely decentralized and distributed fashion, so to reach in finite time a state of static equilibrium in which they cover uniformly the environment. We consider the self-deployment problem in a *ring* (e.g., a circular rim); in particular we investigate under what conditions the problem is solvable by a collection of identical sensors without a global coordinate system, however capable of determining the location (in their local coordinate system) of the other sensors within a fixed distance (called visibility radius). A self-deployment is *exact* if within finite time the distance between any two consecutive sensors along the ring is the same, d; it is ϵ-*approximate* if within finite time the distance between two consecutive sensors is between $d - \epsilon$ and $d + \epsilon$.

We first of all prove that *exact* self-deployment is *impossible* if the sensors do not share a common orientation of the ring. This impossibility result holds even if the sensors have unlimited memory of the past, their visibility radius is unlimited, and all their actions, when active, are instantaneous.

We thus consider the problem in an *oriented* ring. We prove that if the sensors know the desired final distance d, then *exact* self-deployment is *possible*. If the desired final distance d is not known, we prove that ϵ-*approximate* self-deployment is *possible* for any chosen $\epsilon > 0$. The proofs of these results are constructive. In each case we present a simple protocol that allows the sensors to achieve the claimed level of self-deployment. These positive results hold even if sensors are oblivious (i.e., have no memory of past actions and computations), asynchronous (i.e., a sensor becomes active at unpredictable times and the duration of its actions is unpredictable), and have limited visibility radius. Our protocols can be employed, without modifications, on the perimeter of any convex region.

*School of Information Technology and Engineering, University of Ottawa, Canada. (flocchin@site.uottawa.ca)

†Dipartimento di Informatica, UniversitÀ di Pisa, Italy. (prencipe@di.unipi.it)

‡School of Computer Science, Carleton University, Ottawa, Canada. (santoro@scs.carleton.ca)

1 Introduction

1.1 The Framework

We consider a collection of micro-robots or sensors, each capable of limited (sensing, computational) activities, to be deployed in a region ensuring that the area is covered uniformly, so to satisfy some optimization criteria (e.g., to maximize sensing coverage). If the sensors are *mobile*, i.e., capable of moving in the region, they can *self-deploy* without external (e.g., human) assistance.

Some of the initial proposals on the deployment of mobile sensors were still based on centralized approaches, e.g. employing a powerful cluster head to collect the initial location of the mobile sensors and determine their target location [32]. However the current research efforts are on the development of local protocols that allow the sensors to move from an initial random configuration to a uniform one acting in a purely local, decentralized, distributed fashion. An essential requirement is clearly that the sensors will reach a state of *static equilibrium*, that is the self-deployment will be completed within finite time. How this task can be efficiently accomplished continues to be the subject of extensive research (e.g., see [11, 12, 13, 14, 18, 19, 23, 30, 31]). Similar questions have been posed in terms of *scattering* or *coverage* in cooperative mobile robotics and swarm robotics (e.g., [2, 15]), as well as in terms of the *formation* problem for those entities (e.g. [3, 5, 6, 8, 9, 17, 25, 27, 28, 29]). The two key differences are that (1) usually these robots are more powerful (both memory-wise and computationally) than sensors, and (2) typically there is no requirement for the robots to reach a state of static equilibrium (e.g., in most cases the swarm just converges towards a desired formation or pattern). The existing self-deployment protocols differ greatly from each other depending on the assumptions they make; for example some require the sensors to be deployed one at a time [13, 15], while others requires prespecified destinations for the sensors [19]. However, sensors are usually dispersed in the environment all together, more or less at the same time, with no a-priori knowledge of where their final location should be. Actually, unlike the case of ad-hoc networks, for small sensors localization is very hard, so it can not be generally assumed that the sensors know where they are.

The micro-robots we consider here are autonomous (i.e., without a central control), anonymous (i.e., indistinguishable by their appearance), randomly dispersed in the environment, and without a common coordinate system. They are however capable of determining the location (in their local coordinate system) of the other sensors within a fixed radius (called visibility radius). Under these general conditions, none of the existing self-deployment proposals is capable of providing a complete uniform coverage. This impossibility is hardly surprising since those protocols are *generic*, that is they must work in any environment regardless of its topology or structure.

This fact opens a series of interesting questions, first of all whether it is possible for the sensors to self-deploy achieving uniform coverage in specific environments (e.g., corridors, grids, rims). The next important question is on the capabilities and a priori knowledge needed by the sensors to achieve this goal; in other words, how "weak" the sensors can be and still be able to uniformly self-deploy.

Some partial answers have been recently found. In particular, a self-deployment algorithm has recently been developed for the *line* (e.g., a rectilinear corridor) [5], and several have been designed for the *ring* as part of more complex protocols for uniform circle formation [3, 6, 17, 25, 29]. All these protocols yield however only *approximate* solutions; interestingly, they operate even with very weak sensors: anonymous, oblivious, asynchronous, and without a common coordinate system. To date, no *exact* solution exists for these types of sensors.

In this paper we consider precisely these questions and provide a complete answer for these types of sensors in the case of *ring*, that is when the environment where the sensors must be deployed is a circular rim. This situation occurs for example when the the sensors have to surround a dangerous (convex) area and can only move along its outer perimeter.

1.2 Our Results

We study the uniform self-deployment problem in a *ring*: starting from an initial random placement on the ring, the sensors must within finite time position themselves along the ring at (approximately) equal distance; see Figure 1. The sensors are autonomous (i.e., without a central control) and anonymous (i.e., indistinguishable by their appearance). Furthermore, they do not necessarily have a common coordinate system. We assume that each sensor is capable of determining, in its own coordinate system, the position of the sensors within a fixed limited radius, called *visibility radius*.

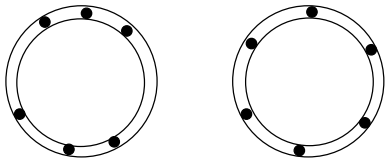


Figure 1: Starting from an initial arbitrary placement (a), the sensors must move to a uniform cover of the ring (b).

A self-deployment algorithm, the same for all sensors, will specify which operations a sensor must perform whenever it is active. We say that a self-deployment algorithm is *exact* if within finite time the sensors reach a *uniform* configuration: the distance between any two consecutive sensors along the ring is the same, d. We say that a self-deployment algorithm is ϵ-*approximate* if the distance between two consecutive sensors is between $d - \epsilon$ and $d + \epsilon$.

We first of all establish a strong negative result. In fact, we prove that *exact* self-deployment is actually *impossible* if the sensors do not share a *common orientation* of the ring; notice that this is much less a requirement than having global coordinates or sharing a common coordinate system. This impossibility result holds even if the sensors (1) have unlimited memory of the past computations and actions, (2) all their actions, when active, are instantaneous and (3) their visibility radius is unlimited.

Faced with this strong negative result, the interesting question becomes under what restrictions the self-deployment problem can be solved with an exact algorithm. Since the impossibility result holds in absence of common orientation of the ring, we consider the problem in *oriented* rings.

We prove that, in an oriented ring, if the sensors know the desired final distance d, then *exact* self-deployment is possible. In fact we present a simple protocol and prove that it allows the sensors to deploy themselves uniformly along the ring in finite time. This positive result holds even for very weak sensors: (1) oblivious (i.e., each sensor has no memory of past actions and computations), (2) asynchronous (i.e., each sensor becomes active at unpredictable times and the duration of its actions is finite but unpredictable), and (3) every sensor has only a fixed visibility radius $v > 2d$.

Finally we turn to the case of an oriented ring when the desired final distance d is unknown. We present another protocol based on a very simple strategy and prove that it is ϵ-approximate for any fixed $\epsilon > 0$. As in [4, 5], the difficulty is not in the protocol but in the proof of its correctness. Also in this case, the protocol works even for the weakest sensors:, oblivious, asynchronous, with only a fixed visibility radius $v \geq 2d$.

Our protocols can be employed not only on a circular rim but also, without modifications, on the perimeter of any convex region.

1.3 Related work

The self-deployment problem has been investigated with the goal to cover the area so to satisfy some optimization criteria, typically to maximize the coverage (e.g., [11, 13, 14, 19, 31]). Typically, distributed self-deployment protocols first discover the existence of coverage holes (the area not covered by any sensor) in the target area based on the sensing service required by the application. After discovering a coverage hole, the protocols calculate the target positions of these sensors, that is the positions where they should move. Loo et al. [19] consider a system consisting of a number of cooperating mobile nodes that move toward a set of prioritized destinations under sensing and communication constraints; unlike them, we do not require prespecified destinations for the sensors. Howard et al. [13] address the problem of incremental deployment, where sensors are deployed one-at-a-time into an unknown environment, and each sensor uses information gathered by previously deployed sensors to determine its deployment location.

The self-deployment problem is related to a well studied problem in the field of swarm robotics: that of the *pattern formation* (e.g., [8, 28]); in particular to the one of *uniform circle formation* [3, 6, 17, 25, 29]. In this problem, very simple robots are required to uniformly place themselves on the circumference of a circle not determined in advance (i.e., the sensors do not know the location of the circle to form). The main difference between these robotics investigations and our self-deployment problem in the ring is that in those problems, the robots can freely move on a two dimensional plane in which they have to form a ring; in contrast, our sensors can move only *on* the ring, which is the entire environment.

A standard assumption in swarm robotics, and used in this paper, is that a sensor is capable of determining the location of its neighbours within its visibility radius. In most

investigations on micro-robots, the determination of one's neighbours is done by sensing capabilities (e.g., vision); in this case, any sensor in the sensing radius is detected even if inactive (e.g. [3, 5, 6, 8, 10, 17, 25, 28]), and thus no other mechanisms are needed. In most investigations on wireless sensor networks, determination of the neighbours within the sensing radius is assumed to be achieved by radio communication (e.g., [26]); in this case, since an inactive sensor does not participate in any communication, the simple activity of determining one's neighbours, to be completed, requires the use of randomization or the presence of sophisticated synchronization and scheduling mechanisms, such as the Virtual Node Layer (e.g., [20, 21, 24]).

In our protocol for unknown d, the strategy we use is *go-to-half*. Interestingly it was shown by Dijkstra [7] that in an unoriented ring *go-to-half* does *not* converge, and hence can not be used even for approximate self-deployment. It does however converge in a *line* as recently proved [5]. Convergence in the unoriented ring has been recently announced for the *go-to-half-half* strategy [6, 25].

2 Terminology and Model

We use the model commonly employed for micro-robots (e.g., [3, 4, 5, 6, 8, 10, 17, 25, 28, 29]). In particular, a sensor (or micro-robot) is viewed as a point and modeled as a computational unit capable of determining the positions of other sensors in its surrounding (within a fixed radius), performing local computations on the determined data, and moving towards the computed destination.

Each sensor has its own local coordinate system and there is no a priori agreement among them; there is however agreement on the unit of distance. The sensors are *autonomous* (i.e., without a central control) and *anonymous*, meaning that they are a priori indistinguishable by their appearance, and they might not have identifiers that can be used during the computation.

Each sensor operates in a *Look - Compute - Move - Wait* cycle: At any point in time, a sensor is either *active* or *inactive*. When *active*, a sensor performs the following operations:

1. (*Locate*) It determines, in its own coordinate system, the positions of the other sensors within its radius of visibility; this constitutes its *view of the world*.

2. (*Compute*) It performs a local computation, according to an algorithm (the same for all sensors) that takes in input its view of the world and returns a destination point.

3. (*Move*) It moves towards the computed destination point; if the destination point is the current location, the sensor stays still.

   A move may stop before the robot reaches its destination, e.g. because of limits to the sensor's motion energy. When *inactive* a sensor

4. (*Wait*) It is idle and does not perform any operation.

There are two limiting assumptions in the model:

(A1) The amount of time required by a sensor to complete a cycle is not infinite, nor infinitesimally small. Note that, as a consequence, each sensor will become active infinitely often.

(A2) The distance traveled by a sensor in a cycle is not infinite, nor infinitesimally small (unless it brings the sensor to the destination point).

Different settings arise from different assumptions that are made on the sensors' capabilities, and on the amount of synchronization among the cycles of the sensors. In particular,

- **Synchronization.** Depending on the amount of synchronization existing among the cycles of the different sensors, two main sub-models are defined, the *semi-synchronous* model (SSYNC), and the *asynchronous* model (ASYNC).

  In the *semi-synchronous* model (SSYNC), the cycles of all sensors are fully synchronized: there is a global clock tick reaching all sensors simultaneously, and a sensor's cycle is an instantaneous event that starts at a clock tick and ends by the next; the only unpredictability is given by the fact that at each clock tick, every sensor is either *active* or *inactive*, and only active sensors perform their cycle. The unpredictability is restricted by the fact that at least one sensor is active at every time instant, and every sensor becomes active at infinitely many unpredictable time instants. This model is used e.g. in [1, 3, 4, 5, 6, 28].

  In the *asynchronous* model (ASYNC), no assumptions on time exist: the amount of time spent in each state of a cycle is finite but otherwise unpredictable. In particular, the sensors do not have a common notion of time. As a result, sensors can be seen by other sensors while moving, and thus computations can be made based on obsolete observations. This (more realistic but more difficult) model is used e.g. in [3, 8, 9, 10, 16, 17, 22].

- **Visibility.** Depending on the location capabilities, two main submodels can be identified, the *limited visibility* model, and the *unnlimited visibility* model.

  In the *unlimited visibility* model, the sensors are capable of determining the location of all sensors regardless of their position in the region. This model is the most commonly used for micro-robots, e.g. in [1, 3, 4, 6, 8, 16, 17, 22, 25, 27, 28].

  In the *limited visibility* model, each sensor can only determine the location of sensors only up to a fixed distance $v > 0$ from it. This (more realistic but more difficult) model is used less often for micro-robots, e.g. in [5, 10, 15], while is most common for wireless sensor networks e.g. in [18, 21, 26].

- **Memory.** In addition to its programs, each sensor has a local memory, or workspace, used for computations and to store different amount of information (e.g., regarding

the location of its neighbours) obtained during the cycles. Two submodels have been identified, depending on whether or not this workspace is persistent.

In the *persistent memory* model, all the information contained in the workspace is *legacy*: unless explicitly erased by the sensor, it will persist thoughout the sensor's cycles. This model is commonly used for both wireless sensor networks and micro-robots. A particular case of persistent memory, sometimes employed for micro-robots, is the *unbounded memory*, where no information is ever erased; hence sensors can remember all past computations and actions (e.g., see [27, 28]).

In the *oblivious* model, model, all the information contained in the workspace is *cleared* at the end of each cycle. In other words, the sensors have *no* memory of past actions and computations, and the computation is based solely on what determined in the current cycle. The importance of obliviousness comes from its link to *self-stabilization* and *fault-tolerance*. This model is used e.g. in [3, 4, 5, 6, 8, 10, 17].

Let $S = \{s_1, \ldots, s_n\}$ be the n sensors initially randomly placed on the ring $\mathcal{C}$ (see Figure 1). We assume that initially no two sensors are placed at the same location; all our algorithms will avoid having two sensors simultaneously occupying the same point.

Let $d_i(t)$ be the distance between sensor s_i and sensor s_{i+1} at time t; when no ambiguity arises, we will omit the time and simply indicate the distance as d_i.

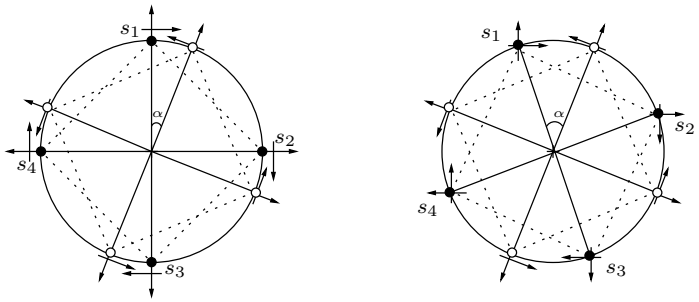
Let $d = L/n$, where L denotes the length of the ring $\mathcal{C}$. We say that the sensors have reached an *exact* self-deployment at time t if $d_i(t) = d$ for all $1 \leq i \leq n$. Given $\epsilon > 0$, we say that the sensors have reached an ϵ-*approximate* self-deployment at time t if $d - \epsilon \leq d_i(t) \leq d + \epsilon$ for all $1 \leq i \leq n$.

We say that an algorithm $\mathcal{A}$ correctly solves the *exact* (resp. ϵ-*approximate*) self-deployment problem if, in any execution of $\mathcal{A}$ by the sensors in $\mathcal{C}$, regardless of their initial position in $\mathcal{C}$, there exists a time t' such that for all $t \geq t'$, the sensors have reached an *exact* (resp. ϵ-*approximate*) self-deployment at time t.

As mentioned in the introduction, we both prove impossibility results and present correct solution protocols. The impossibility results are established even if the sensors are very strong and powerful: they have unlimited memory and unlimited visibility, a situation we denote as Unlim, and their cycles are semi-synchronous. Our self-deployment protocols are designed and proven to work correctly even with very simple weak sensors: they are oblivious and with limited visibility, a situation we denote as Limt, and the cycles are fully asynchronous.

3 Impossibility Without Orientation

In this section, we show that, if the sensors do not share a *common orientation* of the ring, the exact self-deployment problem is unsolvable; that is, if the ring is not oriented, there is *no* deterministic protocol that always allows the sensors to place themselves uniformly on the ring in a finite number of cycles. This result holds even if the sensors are very powerful and they are fully synchronized: the sensors' capabilities are *unlimited* and the scheduling is *semi-synchronous*.



a. b.

Figure 2: (a) An example of starting configuration for the proof of Theorem 1. The black sensors are in S_1, while the white ones in S_2. (b) Theorem 1: the adversary moves only sensors in S_1.

Theorem 1. *Let $s_1, \ldots, s_n$ be all on a ring $\mathcal{C}$. In absence of common orientation of $\mathcal{C}$, there is no deterministic exact self-deployment algorithm even if the sensors' capabilities are* Unlim *and the scheduling is* Ssync.

Proof. By contradiction, let us assume there exists a deterministic algorithm $\mathcal{A}$ that always solves the problem in a finite number of cycles, regardless of the initial position of the sensors in $\mathcal{C}$, and of their individual orientation of the ring. Since the scheduling is Ssync, we can consider each execution as occurring at discrete time steps $t_0, t_1, \ldots$, and it is fully specified once the (non-empty) set of sensors active at each time step is specified.

Let n be even; and let partition the sensors in two sets, $S_1 = \{s_1, \ldots, s_{n/2}\}$ and $S_2 = S \setminus S_1$. The sensors in S_1 and S_2 are placed on the vertices of two regular $n/2$-gons, and the two polygons are rotated of an angle $\alpha < 360°/n$. Furthermore, all sensors have their local coordinate axes rotated so that they all have the same view of the world (refer to Figure 2.a for an example). In other words, the sensors in S_1 share the same orientation, while those in S_2 share the opposite orientation of $\mathcal{C}$. Let us the denote a configuration with such properties by $\mathcal{Y}(\alpha)$. A key property of a $\mathcal{Y}(\alpha)$ configuration is the following.

Claim 3.1. *Let the system be in a configuration $\mathcal{Y}(\alpha)$ at time step t_i.*

1. *If activating only the sensors in S_1,* no *exact self-deployment on $\mathcal{C}$ is reached at time step t_{i+1}, then also activating only the ones in S_2* no *exact self-deployment on $\mathcal{C}$ would be reached at time step t_{i+1}; furthermore, in either case the system would be in a configuration $\mathcal{Y}(\alpha')$ for some $\alpha' < 360°/n$*

2. *If activating only the sensors in S_1 an exact self-deployment on $\mathcal{C}$ is reached at time step t_{i+1}, then also activating only the sensors in S_2 an exact self-deployment on $\mathcal{C}$ would be reached at time step t_{i+1}.*

3. *If activating only the sensors in S_1 an exact self-deployment on $\mathcal{C}$ is reached at time step t_{i+1}, then activating both sets no exact self-deployment on $\mathcal{C}$ would be reached at time step t_{i+1}, and the system would be in a configuration $\mathcal{Y}(\alpha')$ for some $\alpha' < 360°/n$.*

Algorithm 1 The Adversary

(a) If activating only the sensors in S_1 no exact self-deployment on $\mathcal{C}$ is reached: then activate all sensors in S_1, while all sensors in S_2 are inactive; otherwise, activate all sensors. Go to (b).

(b) If activating only the sensors in S_2 no exact self-deployment on $\mathcal{C}$ is reached: then activate all sensors in S_2, while all sensors in S_1 are inactive; otherwise, activate all sensors. Goto (a).

Proof. Cases 1. and 2. immediately follow from the fact that all sensors in S_1 have the same view of the world and the same placement in $\mathcal{C}$ as those in S_2, but with the opposite orientation. Consider now Case 3. Let s_1 be an arbitrary sensor in S_1 (refer to Figure 3). By construction, s_1 has two neighbors on $\mathcal{C}$, s'_2 and s''_2, and both of them are in S_2. Let $\beta = \min(s_1\widehat{c}s'_2, s_1\widehat{c}s''_2)$ (clearly, $s_1\widehat{c}s'_2$ cannot be equal to $s_1\widehat{c}s''_2$, otherwise the sensors would be uniformly placed on $\mathcal{C}$). By hypothesis, by activating only the sensors in S_1, the sensors would reach an exact self-deployment on $\mathcal{C}$. In other words, they would all rotate of an angle γ so that, at time t_{i+1}, $\beta + \gamma = 360°/n$. Symmetrically, if only the sensors in S_2 would be activated, they would rotate of an angle δ so that, at time t_{i+1}, $\beta + \delta = 360°/n$. Therefore, since $\beta + \gamma + \delta \neq 360°/n$, by activating all sensors, an uniform placement on $\mathcal{C}$ will not be reached at time t_{i+1}. Furthermore, by activating all sensors, at time t_{i+1} the sensors in S_1 and S_2 would be placed on the vertices of two regular $n/2$-gons, the two polygons are rotated of an angle $\alpha' < 360°/n$, and all sensors still have the same view of the world. □

Let now continue the proof of the theorem. In the following, we define an Adversary that will force $\mathcal{A}$ to never succeed in solving the problem. Algorithm 1 describes the protocol followed by the Adversary. The adversary will choose $\mathcal{Y}(\alpha)$ as the initial configuration. By Claim 3.1, if the configuration at time $t_i \geq t_0$ is $\mathcal{Y}(\alpha)$ for some $\alpha < 360°/n$, then regardless of whether the Adversary executes step (a) or (b), the resulting configuration is $\mathcal{Y}(\alpha')$ for some $\alpha' < 360°/n$, and hence *no* exact self-deployment on $\mathcal{C}$ is reached at time step t_{i+1}. Hence, there exists an infinite execution of $\mathcal{A}$ in which no exact self-deployment will ever be reached. The alternating between steps (a) and (b) by the Adversary ensures the feasibility of this execution: every sensor will in fact become active infinitely often. Hence, a contradiction with the correctness of $\mathcal{A}$ is obtained. □

Since the impossibility result of Theorem 1 holds in absence of common orientation of the ring, we will now focus on *oriented* rings; we will then consider two cases, depending on whether or not the desired final distance d is known to the sensors.

4 Oriented Ring with Interdistance Known

Let the sensors share a common orientation of the ring. In this section we examine the case when the desired final distance d is known or computable (e.g., both the number or sensors

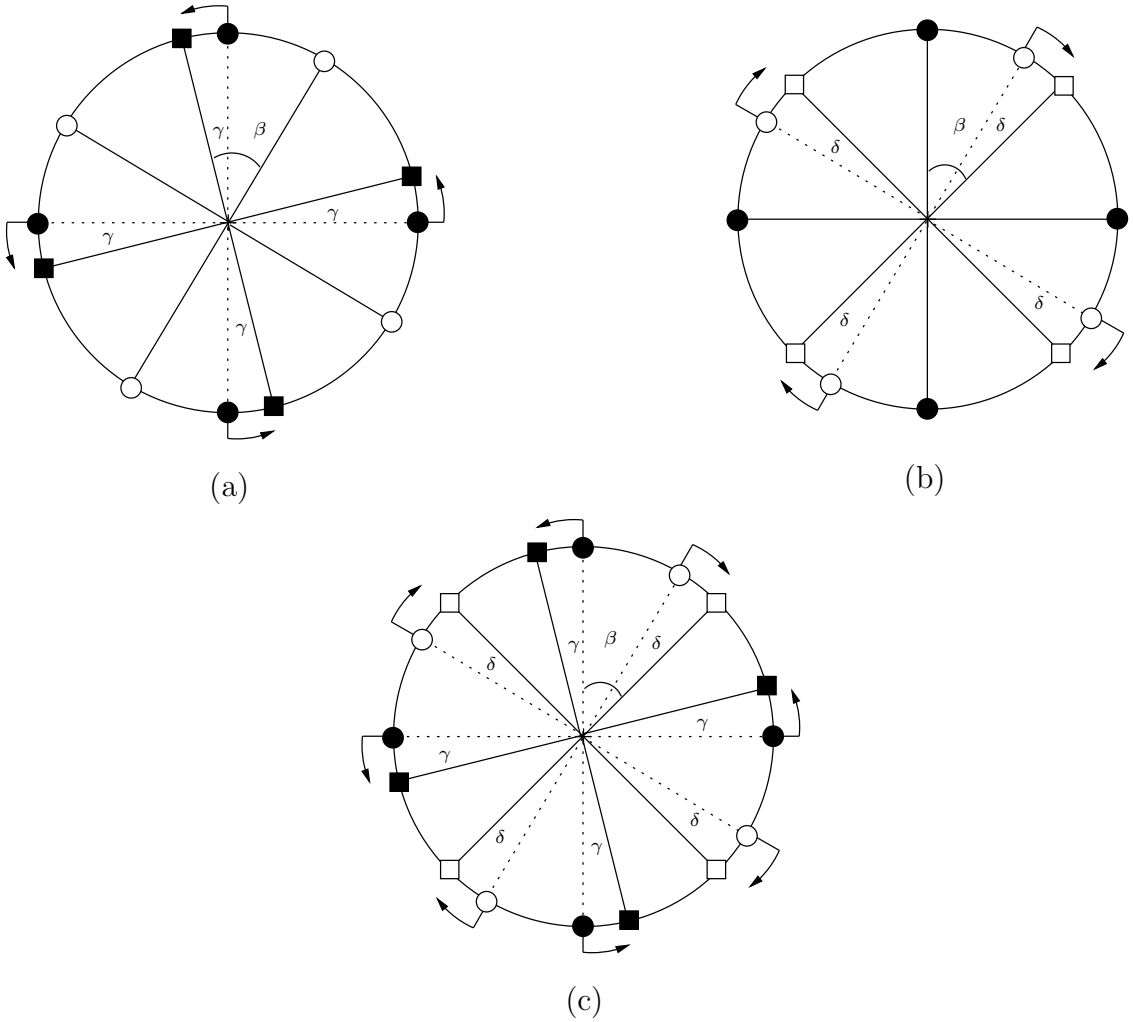


Figure 3: Theorem 1. (a) If only the sensors in S_1 are activated at t, all sensors would be uniformly placed at time $t+1$, with $\beta+\gamma=45°$. (b) If only the sensors in S_2 are activated at t, all sensors would be uniformly placed at time $t+1$, with $\beta+\gamma=45°$. (c) Therefore, if all sensors would be activated at t, they would not be in an exact self-deployment on $\mathcal{C}$, having $\gamma+\beta+\delta \neq 2\pi/n = 45°$. In all figures, the squares represent the destination of the active sensors.

and the length of the ring are known). We prove that, in this case, *exact* self-deployment is indeed possible. This positive result holds even with weak asynchronous sensors, provided their visibility radius is at least $2d$.

Theorem 2. *Let $s_1, \ldots, s_n$ share a common orientation of the ring $\mathcal{C}$, and be able to locate to distance $2d$. If they know d, then* exact *self-deployment is possible even if the sensors' capabilities are* LIMT *and the scheduling is* ASYNC.

The proof of Theorem 2 is constructive: we present a simple protocol and prove that, under the theorem hypothesis, allows asynchronous sensors with limited capabilities to deploy themselves uniformly along the ring in finite time.

4.1 The Algorithm

The algorithm is very simple: sensors asynchronously and independently observe clockwise at distance $2d$, then they position themselves at distance d from the closest observed sensor (if any).

Protocol UNIFORM KNOWN (for sensor s_i)

- Locate clockwise at distance $2d$. Let d_i be the distance to s_{i+1} (if visible, else $d_i = 2d$).
- If $d_i \leq d$ do not move.
- If $d_i > d$ move clockwise and place yourself at distance d from s_{i+1} (if visible, else at distance d from current location).

4.2 Correctness

We say that a sensor is *white* if its distance to the clockwise neighbor is greater than or equal to d. We say that a sensor is *gray* if such a distance is smaller than d. Moreover we say that a white sensor is *good* if its distance to the clockwise neighbor is exactly d, it is *large* if its distance is strictly greater than d.

To prove that the algorithm is correct, we must prove that, within finite time, all sensors become *good*.

We call a *white bubble* a sequence of consecutive white sensors delimited by grey sensors. Let $W = s_i, s_{i+1}, \ldots, s_{i+m}$ be a white bubble. Sensor s_{i-1} is said to be the predecessor of the bubble, sensor s_{i+m+1} is the successor. Clearly predecessors and successors of a white bubble are gray, unless the ring contains white sensors only; notice that in this case all sensors are good. The size of W , indicated as $|W|$ is the number of white sensors composing the bubble (in this example m), its length, indicated by $l(W)$, is the length of the ring between the predecessor of the white bubble and its successor (assuming not all sensors are white); i.e., $l(W) = \sum_{j=-1}^{m} d_{i+j}$. Similarly, we define a *gray bubble* $G = s_i, s_{i+1}, \ldots, s_{i+m}$ as a sequence

of consecutive gray sensors delimited by white sensors. Its size $|G|$ is the number of gray sensors in G; the length $l(G)$ is defined as the length of the ring between the first and the last gray sensor in G (note that this definition is different from $l(W)$).

The next two lemmas contain some simple facts.

Lemma 1. *At each point in time, if there are gray sensors, then the number of white bubbles equals the number of gray bubbles.*

Lemma 2. *At each point in time, if there are grey sensors there must be at least a bubble (i.e., a large sensor).*

Lemma 3. *A white sensor cannot become gray.*

Proof. In order for a white sensor s_j to become gray, its distance to the next sensor s_{j+1} should become smaller than d. By definition, sensors move clockwise and move according to the algorithm; so sensor s_{j+1} will never get closer to s_j. On the other hand, by definition of our algorithm, sensor s_j will never move at a distance smaller than d to s_{j+1}. □

Lemma 4. *Let $W = s_i, s_{i+1}, \ldots s_{i+m}$ be a white bubble in the ring at time t. If $l(W) \geq d \cdot (|W| + 1)$, in finite time, say at time t', the size of the bubble increases.*

Proof. We want to prove by induction on the sensors in W that, by time t', all sensors in the white bubble are good, and the predecessor s_{i-1} is white (which means that the bubble has become bigger).

By definition of our algorithm, in finite time, say at time t_1, s_{i+m} becomes good placing itself at distance d to the successor of W. Let us assume that at time $t_j < t'$ all sensors $s_{i+m}, s_{i+m-1} \ldots, s_{i+m-j}$ are good. Let us consider now sensor $s_{i+m-j-1}$. If this sensor is not already good, by definition of the algorithm and since by hypothesis the successor of W does not become white, $s_{i+m-j-1}$ will move to place itself at distance d to s_{i+m-j}, thus becoming good at time t_{j+1}.

Thus, in finite time, say at t', all sensors in the bubble are good, which means that the distance between sensor s_i and sensor s_{i+m+1} is equal to $d \cdot m = d \cdot s(W)$. Since, by hypothesis, $l(W) \geq d \cdot (s(W) + 1)$,it follows that the distance between s_{i-1} and s_i becomes greater than or equal to d, which means that s_{i-1} has become white. □

Lemma 5. *Let $W_1, \ldots W_z$ be the white bubbles present in the ring at time t. At least one of these bubble W_k is such that $l(W_k) \geq d \cdot |W_k| + 1$.*

Proof. By contradiction, let $l(W_i) < d \cdot (|W_i| + 1)$, for all W_i. The length L of the ring is the sum of the lengths of all white bubbles and all gray bubbles. That is, from Lemma 1, $L = \sum_{i=1}^{z}(l(W_i) + l(G_i))$. By hypothesis, $\sum_{i=1}^{z} l(W_i) < d \sum_{i=1}^{z} |W_i| + d \cdot z$. Moreover, by definition of gray bubble, $\sum_{i=1}^{z} l(G_i) < d \sum_{i=1}^{z}(|G_i| - 1) = d \sum_{i=1}^{z} |G_i| - d \cdot z$. Summing up, we have $L < d \sum_{i=1}^{z}(|G_i| + |W_i|) = d \cdot n$, a contradiction. □

By Lemmas 4 and 5, we have that:

Lemma 6. *The number of grey sensors decreases.*

Finally, by Lemmas 3 and 6 the correctness of the algorithm follows.

Theorem 3. *In finite time all sensors are* good.

In other words, within finite time, the sensors have performed an exact self-deployment; thus, observing that the algorithm operates within LIMT and ASYNC, the claim of Theorem 2 holds.

5 Oriented Ring with Interdistance Unknown

In this section we examine the case when the sensors share a common orientation of the ring, but the desired final distance d is not known nor computable. We prove that, in this case, ϵ-*approximate* self-deployment is indeed possible for any ϵ. This positive result holds even with weak asynchronous sensors, provided their visibility radius is greater than $2d$.

Theorem 4. *Let $s_1, \ldots, s_n$ share a common orientation of the ring $\mathcal{C}$, and be able to locate to distance $v > 2d$. Then* ϵ-approximate *self-deployment is possible even if the sensors' capabilities are* LIMT *and the scheduling is* ASYNC.

Also in this case the proof is constructive: we present a simple protocol and prove that, under the theorem hypothesis, allows asynchronous sensors with limited capabilities to deploy themselves uniformly along the ring in finite time.

5.1 The Algorithm

Also this algorithm is very simple: sensors asynchronously and independently locate in both directions at distance v, then they position themselves in the middle between the closest observed sensor (if any).

Protocol UNIFORM UNKNOWN (for sensor s_i)

- Locate around at distance v. Let d_i be the distance to next sensor, d_{i-1} the distance to the previous (if no sensor is visible clockwise, $d_i = v$, analogously for counterclockwise).
- If $d_i \leq d_{i-1}$ do not move.
- If $d_i > d_{i-1}$ move to $\frac{d_i + d_{i-1}}{2} - d_{i-1}$ clockwise.

5.2 Correctness

Let $d_{min}(t) = Min\{d_i(t)\}$ and $d_{max}(t) = Max\{d_i(t)\}$. Let C be the length of the ring. First observe the following simple fact:

Lemma 7. *We have that:* $\forall t$, $d_{min}(t) \leq d$ *and* $d_{max}(t) \geq d$.

Proof. By contradiction. Let the minimum distance be greater than d. We would have that $C > k \cdot d$, which is impossible since by definition $C = k \cdot d$. Same argument holds for d_{max}. □

The next lemma shows that if, at some point there is a unique minimum (resp. maximum) interval, it will become bigger (resp. smaller).

Lemma 8. *If at time* t *there is a unique minimum interval, we have that:* $\forall t, \exists t' > t : d_{min}(t') > d_{min}(t)$. *If at time* t *there is a unique maximum interval, we have that:* $\forall t, \exists t' > t : d_{max}(t') < d_{max}(t)$.

Proof. Let s_{j-1} and s_j be the sensors that delimit the minimum interval $[s_{j-1}, s_j]$, whose length is $d_{j-1}(t) = d_{min}(t)$ at time t. First observe that, since $d_{j-2}(t) > d_{j-1}(t)$, by the algorithm we know that sensor s_{j-1} does not move at time t; actually, it will not be able to move as long as d_{j-2} remains greater than d_{j-1} (i.e., as long as s_j does not move). Consider now the first time t' when s_j is activated. Since s_{j-1} has not moved from time t to time t', we have that, at time t', $d_{j-2}(t')$ is still greater than $d_{j-1}(t')$. At time t', s_i then moves following the rule of the algorithm and $d_{j-1}(t') = \frac{d_{j-1}(t)+d_j(t')}{2} \geq \frac{d_{j-1}(t)+d_j(t)}{2} > d_{j-1}(t)$. Similar argument holds for d_{max}. □

We now show that if at some point there are several minimum (resp. maximum) intervals of a certain length, their number will decrease.

Lemma 9. *If at time* t *there are* $r > 1$ *minimum intervals of length* $d_{min}(t)$, *either all intervals have length* d *and the sensors are deployed, or there exists a time* $t' > t$ *when the number of minimum intervals of length* $d_{min}(t)$ *is* $r' < r$.

Proof. First notice that, if at time t a sensor s_j delimiting a minimum interval $[s_{j-1}, s_j]$ is activated, it will not move if $d_{min}(t) = d_{j-1}(t) = d_j(t)$ (i.e., if $[s_j, s_{j+1}]$ is another minimum interval), it will instead move if $d_{j-1}(t) < d_j(t)$.

Consider the first time t' when a sensor s_j delimiting a minimum interval $[s_{j-1}, s_j]$, which is *not* followed by another minimum interval, is activated. Notice that such a sensor must exist otherwise we would be in a situation when all sensors are deployed at distance d from each other. In this case we know that at time t' there are still at most s minimum intervals and that $d_{j-1}(t') < d_j(t')$. Sensor s_j then moves and $d_{j-1}(t') = \frac{d_{j-1}(t)+d_j(t')}{2} \geq \frac{d_{j-1}(t)+d_j(t)}{2} > d_{j-1}(t)$, thus it is not minimum anymore and the number of minimum intervals is now strictly smaller than r. □

Analogously,

Lemma 10. *If at time* t *there are* $r > 1$ *maximum intervals, either all intervals have length* d *and the sensors are deployed, or there exists a time* t' *when the number of maximum intervals is* $r' < r$.

We now show that the minimum intervals converge to a value $A = d - \gamma_{min}$, with $\gamma_{min} \geq 0$, and the maximum intervals converge to a value $B = d + \gamma_{min}$, with $\gamma_{max} \geq 0$.

Lemma 11. *Let $d_{min}(t)$ (resp $d_{max}(t)$) be the distance of a minimum (resp. maximum) interval at time t. We have that, for any arbitrary small $\epsilon > 0$ there exists a time $t' > t$ such that, $\forall t'' > t'$: $|d_{min}(t'') - A| \leq \epsilon$, and, $\forall t'' > t'$: $|d_{max}(t'') - B| \leq \epsilon$.*

Proof. From Lemmas 8 and 9 the intervals must converge; from Lemma 7 the minimum must converge to a value smaller than (or equal to) d, and the maximum must converge to a value greater than (or equal to) d. □

Let us call *A-regular* at time t an interval that, at time t is ϵ-close to A; that is an interval whose length $d_j(t)$ is such that $|d_j(t) - A| \leq \epsilon$. Analogously, we call *B-regular* an interval that is ϵ-close to B. We call *A-irregular* at time t an interval that, at time t, is smaller than d, but not ϵ-close to A; *B-irregular* one that is greater than d, but not ϵ-close to B.

The following lemma shows that there exists a time t, after the time when the previous Lemma 11 holds, when any interval greater than the minimum (and smaller than d) is *A-regular*, and any interval smaller than the maximum (and greater than d) is *B-regular*. In other words, each interval is either ϵ-close to A or to B. Notice that this property is not obvious; in fact, the only thing we know up to now is the convergence to A and B of the minimum/maximum intervals over time, while nothing is known about the other intervals.

Lemma 12. *Let $\epsilon > 0$ be arbitrarily small, and let t'_ϵ be a time when Lemma 11 holds. There exists a time $t''_\epsilon > t'_\epsilon$ when: for all intervals $[s_j, s_{j+1}]$ with $d_j(t'') \leq d$, $|d_j(t''_\epsilon) - A| \leq \epsilon$; for all intervals $[s_i, s_{i+1}]$ with $d_i(t''_\epsilon) \geq d$, $|d_i(t'') - B| \leq \epsilon$.*

Proof. By contradiction, assume such a situation never happens. Then, there must exist a time t when there are both *A-irregular* and *B-irregular* intervals.

Consider the following execution: 1) if there are *A-regular* intervals followed by *B-regular* intervals, let the sensors between them move. Notice that whenever a sensor between a *A-regular* and a *B-regular* intervals move, both intervals become irregular. Further notice that, after this activation rule, we are guaranteed that a sequence of regular intervals delimited by irregular intervals contains only intervals of the same type (*A-regular* or *B-regular* only). 2) Consider any *A-irregular* interval $[s_j, s_{j+1}]$. Let it be preceded by $k \geq 0$ *A-regular* intervals (delimited by sensors $s_{j-1} \dots s_{j-k}$) and followed by $h \geq 0$ *B-regular* interval (delimited by sensors $s_{j+2} \dots s_{j+h-1}$). Activate sensors $s_{j+1}, s_{j+2} \dots s_{j+h-1}$, $s_{j-1} \dots s_{j-k}$, in this order. It is easy to see that their movement transforms all those interval in irregular intervals. 3) Apply the same schedule to all *B-irregular* intervals (preceded by *A-regular* intervals and followed by *B-regular* intervals).

Notice that, by the above activation rules, a sequence of *A-regular* intervals becomes irregular if it is followed by *B-irregular* intervals or if it is preceded by *A-regular* intervals. Thus, after the activation rules of 2) and 3) we are in a situation where all intervals (included the minimum) are irregular and thus Lemma 11 is violated. □

Lemma 13. *Let t be a time when Lemma 12 holds. If at some time $t' > t$ at least an interval becomes irregular, then there exists a time $t'' > t'$ when all intervals are irregular.*

Proof. The argument is very similar to the one of Lemma 12. □

We now show that, after a time when Lemma 12 holds, all intervals actually converge to d (i.e., $A = B = d$).

Lemma 14. *Let $\epsilon > 0$ be arbitrarily small, and let t'_ϵ be a time when Lemma 12 holds. If $B - A > 2\epsilon$, at least an interval becomes irregular.*

Proof. Let $t_1 = t'_\epsilon$. We will show that, under the conditions of the statement there exists a movement of a sensor at time t_1 that create an irregular interval.

Consider two consecutive intervals $[s_i, s_{i+1}]$ and $[s_{i+1}, s_{i+2}]$ such that $d_i(t_1) < d$ and $d_{i+1}(t_1) > d$. Such intervals must exist because otherwise all the sensors would be deployed at precisely distance d from each other. By Lemma 12, we have that:

$$|d_i(t_1) - A| \leq \epsilon \tag{1}$$

$$|d_{i+1}(t_1) - B| \leq \epsilon \tag{2}$$

Let sensor s_{i+1} move at time t_1. As a result of the movement, at any time $t_2 > t_1$ before any other movement of the sensors, we have that:

$$d_i(t_2) = \frac{d_i(t_1) + d_{i+1}(t_1)}{2} = d_{i+1}(t_2) \tag{3}$$

We now consider several different cases.

Case 1. $A + \epsilon \geq d_i(t_1) > A$ and $B + \epsilon \geq d_{i+1}(t_1) > B$. From Equation 3 and for the assumption, we have that:

$$\frac{A+B}{2} < d_i(t_2) = d_{i+1}(t_2) \leq \frac{A+B+2\epsilon}{2} \tag{4}$$

We now consider the two case $d_i(t_2) > d$ and $d_i(t_2) < d$ and in both we will derive a contradiction.

1.1) Let $d_i(t_2) > d$. In this case we would have that $\frac{A+B+2\epsilon}{2} \geq d_i(t_2) > d$. We now consider the two cases: $d_i(t_2) > B$, and $d_i(t_2) < B$. If $d_i(t_2) > B$ it must be that $\frac{A+B+2\epsilon}{2} \geq B$, which would imply $A + 2\epsilon > B$, which is a contradiction with the assumption that $B - A > 2\epsilon$. It follows that $d < d_i(t_2) < B$. However, from Equation 2, we must have that $B - d_i(t_2) \leq \epsilon$, which would imply $B - \frac{A+B+2\epsilon}{2} \leq \epsilon$, that is $B - A \leq \epsilon$, which is a contradiction.

1.2) Let $d_i(t_2) < d$. In this case we would have to show that, by Equation 1, $d_i(t_2) - A \leq \epsilon$. However, $d_i(t_2) - A > \frac{B}{2} - \frac{A}{2}$, which is clearly greater than ϵ. Contradiction.

Case 2. $d_i(t_1) < A$ and $d_{i+1}(t_1) < B$. From Equation 3 and for the assumption, we have that: $d_i(t_2) = d_{i+1}(t_2) < \frac{A+B}{2}$.

By Equations 1 and 2 we must have that $A - d_i(t_1) \leq \epsilon$ and $B - d_{i+1}(t_1) \leq \epsilon$. In other words, $d_i(t_1) \geq A - \epsilon$, and $d_{i+1}(t_1) \geq B - \epsilon$. By Equation 3 and by the above, we have that $d_i(t_2) \geq \frac{A+B-2\epsilon}{2}$ (notice that, since $B > A$, this implies that $d_i(t_2) > A$). Thus we have:

$$\frac{A+B-2\epsilon}{2} \leq d_i(t_2) < \frac{A+B}{2} \tag{5}$$

Consider now the two possibilities $A < d_i(t_2) < d$ and $d_i(t_2) > d$: in both cases, we will show a contradiction.

2.1) If $A < d_i(t_2) < d$, Equation 1 must hold, that is $d_i(t_2) - A \leq \epsilon$. However, $d_i(t_2) - A \geq \frac{B}{2} - \frac{A}{2} - \epsilon$, which is clearly greater than ϵ, since $B - A > 2\epsilon$.

2.2) Consider now the case $d_i(t_2) > d$, in this case, by Equation 2, we must have $|d_i(t_2) - B| \leq \epsilon$. Since $A < B$, and thus $\frac{A+B}{2} < B$, we have that $d_i(t_2) < B$, so, by Equation 2 it must be: $B - d_i(t_2) \leq \epsilon$, or, in other words, $d_i(t_2) \geq B - \epsilon$. However from Equation 5, we know that $d_i(t_2) < \frac{A}{2} + \frac{B}{2}$ which is clearly smaller than $B - \epsilon$ (because $B - A > 2\epsilon$). Contradiction.

Case 3. $A + \epsilon \geq d_i(t_1) > A$ and $d_{i+1}(t_1) < B$. We have $d_i(t_1) > A$, and by definition we have $B - d_{i+1}(t_1) \leq \epsilon$; thus, from Equation 3 we obtain: $d_i(t_2) = d_{i+1}(t_2) \geq \frac{A+B-\epsilon}{2}$. Moreover, by the assumptions we get $d_i(t_2) = d \leq \frac{A+B+\epsilon}{2}$. Thus

$$\frac{A+B-\epsilon}{2} \leq d_i(t_2) < \frac{A+B+\epsilon}{2} \tag{6}$$

3.1) If $d_i(t_2) < d$ we should have (by Equation 1) that $d_i(t_2) - A \leq \epsilon$. However, by Equation 6, we have $d_i(t_2) - A \geq \frac{B-A-\epsilon}{2} \geq \epsilon$. Contradiction.

3.2) Let $d_i(t_2) > d$. First observe that $d_i(t_2)$ cannot be greater than B because we have $d_i(t_2) \leq \frac{A+B+\epsilon}{2} < B$); thus $d_i(t_2) < B$. We should have (by Equation 2) that $B - d_i(t_2) \leq \epsilon$. However, from Equation 6 we know that $d_i(t_2) - B \leq \frac{A+B+\epsilon}{2} < \epsilon$. Contradiction.

Case 4. $d_i(t_1) < A$ and $B + \epsilon \geq d_{i+1}(t_1) > B$. We have $d_i(t_1) < A$ and $d_{i+1}(t_1) \leq B + \epsilon$; thus, from Equation 3 we obtain: $d_i(t_2) = d_{i+1}(t_2) < \frac{A+B+\epsilon}{2}$. Moreover, by assumption $d_{i+1}(t_1) \leq B + \epsilon \geq$, and by definition $A - d_{i+1}(t_1) \leq \epsilon$, so we get: $d_i(t_2) \geq \frac{A+B-\epsilon}{2}$. Thus

$$\frac{A+B-\epsilon}{2} \leq d_i(t_2) < \frac{A+B+\epsilon}{2} \tag{7}$$

The rest of the proof proceeds like for Case 3. □

Theorem 5. *For any arbitrary small $\epsilon > 0$ there exists a time t, such that $\forall t' > t, \forall i$: $|d_i(t') - d| \leq \epsilon$.*

Proof. By contradiction. Let $A \neq B$. From Lemma 12, there is a time t when all intervals are ϵ-close to A and B. From Lemma 14, at least one interval will become irregular at some time $t' > t$. However, by Lemma 13 there is a time $t'' > t'$ when all intervals become irregular (including the minimum and the maximum). This contradicts Lemma 11. □

In other words, within finite time, the sensors have performed a ϵ-approximate self-deployment; thus, observing that the algorithm operates within LIMT and ASYNC, the claim of Theorem 4 holds.

6 Conclusions

In this paper we have provided a strong characterization of the self-deployment problem of a mobile sensor network in a ring. In particular, we have shown that exact self-deployment of powerful sensor is unsolvable even under a SSYNC scheduler if the sensors do not share a common orientation of the ring; and we have presented the first provably correct exact self-deployment solution that works also when the sensors are limited and asynchronous, provided the final distance d is known. In the case when the ring is oriented but d is not known, we have presented a simple protocol that achieves ϵ-approximate self deployment for any $\epsilon > 0$.

From a theoretical point of view, the results of this paper, together with the existing ones for the line [5], are the first steps in understanding the computational nature (i.e., limitations and properties) of the self-deployment problem for mobile sensor networks in constrained environments. From a practical point of view, we have provided protocols that are simple, provably correct, and easily implementable; they can be executed by very weak sensors; and they can be employed along the border of any convex region.

Several research questions are still open. The foremost open problem is the determination of whether knowledge of d is indeed necessary for exact self-deployment in an oriented ring. Should this be the case, the natural open problem is to determine which is the "weakest" additional assumption (e.g., a priori knowledge, capability) that would make exact self-deployment possible.

A more general and challenging open problem is to find additional sensors' capabilities that would enable the existence of an asynchronous exact self-deployment protocol in un-oriented rings. Another important research direction is to identify meaningful efficiency parameters and study the complexity as well the computability of the problem.

Acknowledgments. The authors would like to thank Vincenzo Gervasi, Toni Mesa, and Linda Pagli for the many discussions. This work is supported in part by the Natural Sciences and Engineering Research Council of Canada.

References

[1] N. Agmon, D. Peleg, Fault-tolerant gathering algorithms for autonomous mobile robots, In *Proc. of the 15th ACM-SIAM Symposium on Discrete Algorithms*, pages 1070–1078, 2004.

[2] Y. U. Cao, A. S. Fukunaga, A. B. Kahng, and F. Meng. Cooperative Mobile Robotics: Antecedents and Directions. In *Proc. IEEE/TSJ International Conference on Intelligent Robots and Systems*, pages 226–234, 1995. Yokohama, Japan.

[3] I. Chatzigiannakis, M. Markou, and S. Nikoletseas. Distributed Circle Formation for Anonymous Oblivious Robots. In *Experimental and Efficient Algorithms: Third International Workshop (WEA 2004)*, volume LNCS 3059, pages 159 –174, 2004.

[4] R. Cohen and D. Peleg. Convergence Properties of the Gravitational Algorithm in Asynchronous Robot Systems. In *Proc. of the 12th European Symposium on Algorithms*, volume LNCS 3221, pages 228–239, 2004.

[5] R. Cohen and D. Peleg. Local Algorithms for Autonomous Robot Systems. In *Proc. of the 13th Colloquium on Structural Information and Communication Complexity*, volume LNCS 4056, pages 29–43, 2006.

[6] X. Défago and A. Konagaya. Circle Formation for Oblivious Anonymous Mobile Robots with No Common Sense of Orientation. In *Workshop on Principles of Mobile Computing*, pages 97–104, 2002.

[7] E. W. Dijkstra. *Selected Writings on Computing: A Personal Perspective*, pages 34–35. Springer, New York, 1982.

[8] P. Flocchini, G. Prencipe, N. Santoro, and P. Widmayer. Hard Tasks for Weak Robots: The Role of Common Knowledge in Pattern Formation by Autonomous Mobile Robots. In *Proc. 10th International Symposium on Algorithm and Computation*, pages 93–102, 1999.

[9] P. Flocchini, G. Prencipe, N. Santoro, and P. Widmayer. Pattern Formation by Autonomous Robots Without Chirality. In *Proc. 8th Int. Colloquium on Structural Information and Communication Complexity*, pages 147–162, June 2001.

[10] P. Flocchini, G. Prencipe, N. Santoro, and P. Widmayer. Gathering of Asynchronous Mobile Robots with Limited Visibility. *Theoretical Computer Science*, 337:147–168, 2005.

[11] N. Heo and P. K. Varshney. A Distributed Self Spreading Algorithm For Mobile Wireless Sensor Networks. In *Proceedings IEEE Wireless Communication and Networking Conference*, volume 3, pages 1597–1602, 2003.

[12] N. Heo and P. K. Varshney. Energy-efficient Deployment of Intelligent Mobile Sensor Networks. *IEEE Transactions on Systems, Man, and CyberNetics - Part A: Systems and Humans*, 35(1):78–92, 2005.

[13] A. Howard, M. J. Mataric, and G. S. Sukhatme. An Incremental Self-deployment Algorithm for Mobile Sensor Networks. *Autonomous Robots*, 13(2):113–126, 2002.

[14] A. Howard, M. J. Mataric, and G. S. Sukhatme. Mobile Sensor Network Deployment Using Potential Fields: A Distributed, Scalable Solution to The Area Coverage Problem. In *Proceedings of the 6th International Symposium on Distributed Autonomous Robotics Systems (DARS'02)*, pages 299–308, 2002.

[15] T.-R. Hsiang, E. Arkin, M. A. Bender, S. Fekete, and J. Mitchell. Algorithms for rapidly dispersing robot swarms in unknown environments. In *Proceedings 5th Workshop on Algorithmic Foundations of Robotics (WAFR)*, pages 77–94, 2002.

[16] Y. Katayama, Y. Tomida, H. Imazu, N. Inuzuka, K. Wada. Dynamic Compass Models and Gathering Algorithms for Autonomous Mobile Robots In *14th Colloquium on Structural Information and Communication Complexity*, 2007.

[17] B. Katreniak. Biangular Circle Formation by Asynchronous Mobile Robots. In *Proc. of the 12th Int. Colloquium on Structural Information and Communication Complexity*, pages 185–199, 2005.

[18] X. Li and N. Santoro. An Integrated Self-deployment and Coverage Maintenance Scheme for Mobile Sensor Networks. In *Proc of 2nd Int. Conf. on Mobile Ad-Hoc and Sensors Networks (MSN'06)*, 2006.

[19] L. Loo, E. Lin, M. Kam, and P. Varshney. Cooperative Multi-Agent Constellation Formation Under Sensing and Communication Constraints. *Cooperative Control and Optimization*, pages 143–170, 2002.

[20] N. Lynch, S. Mitra, and T. Nolte. Motion coordination using virtual nodes. In *Proc. of the 44th IEEE Conference on Decision and Control*, 2005.

[21] S. E. Nikoletseas. Models and algorithms for wireless sensor networks (smart dust). In *Proc. 32nd Conference on Current Trends in Theory and Practice of Computer Science*, pages 64–83, 2006.

[22] Y. Oasa, I. Suzuki, M. Yamashita, A Robust Distributed Convergence Algorithm for Autonomous Mobile Robots. In *Proc. IEEE Int. Conf. on Systems, Man and Cybernetics*, pages 287–292, 1997.

[23] S. Poduri and G.S. Sukhatme. Constrained Coverage for Mobile Sensor Networks. In *Proc. IEEE Int. Conference on Robotic and Automation*, pages 165–173, 2004.

[24] O. Powell, P. Leone, and J. Rolim. Energy optimal data propagation in wireless sensor networks. *Journal of Parallel and Distributed Computing*, 67(3):302–317, 2007.

[25] S. Samia, X. Défago, and T. Katayama. Convergence Of a Uniform Circle Formation Algorithm for Distributed Autonomous Mobile Robots. In *In Journés Scientifiques Francophones (JSF), Tokio, Japan*, 2004.

[26] I. Stojmenovic. *Handbook of Sensor Networks: Algorithms and Architectures*. Wiley, 2005.

[27] K. Sugihara and I. Suzuki. Distributed Algorithms for Formation of Geometric Patterns with Many Mobile Robots. *Journal of Robotics Systems*, 13:127–139, 1996.

[28] I. Suzuki and M. Yamashita. Distributed Anonymous Mobile Robots: Formation of Geometric Patterns. *SIAM J. Computing*, 28(4):1347–1363, 1999.

[29] O. Tanaka. Forming a Circle by Distributed Anonymous Mobile Robots. Technical report, Department of Electrical Engineering, Hiroshima University, Hiroshima, Japan, 1992.

[30] G. Wang, G. Cao, and T. La Porta. A Bidding Protocol for Deploying Mobile Sensors. In *Proceedings of 11th IEEE International Conference on Network Protocols*, pages 315–325, 2003.

[31] G. Wang, G. Cao, and T. La Porta. Movement-assisted Sensor Deployment. In *Proceedings of IEEE INFOCOM*, volume 4, pages 2469–2479, 2004.

[32] Y. Zou and K. Chakrabarty. Sensor Deployment and Target Localization in Distributed Sensor Networks. *ACM Transactions on Embedded Computing Systems*, 3(1):61–91, 2004.