Alcom-FT Technical Report Series
ALCOMFT-TR-03-163

Distributed Circle Formation for Anonymous Oblivious Robots *

Ioannis Chatzigiannakis † Michael Markou ‡ Sotiris Nikoletseas†‡

July 31, 2003

Abstract

This paper deals with systems of multiple mobile robots each of which observes the positions of the other robots and moves to a new position so that eventually the robots form a circle. In the model we study, the robots are anonymous and oblivious, in the sense that they cannot be distinguished by their appearance and do not have a common x-y coordinate system, while they are unable to remember past actions.

We propose a new distributed algorithm for circle formation on the plane. We prove that our algorithm is correct and provide an upper bound for its performance. In addition, we conduct an extensive and detailed comparative simulation experimental study with the DK algorithm described in [7]. The results show that our algorithm is very simple and takes considerably less time to execute than algorithm DK.

1 Introduction, Our Results and Related Work

Lately, the field of cooperative mobile robotics has received a lot of attention from various research institutes and industries. A focus of these research and development activities is that of distributed motion coordination, since it allows the robots to form certain patterns and move in formation towards cooperating for the achievement of certain tasks. Motion planning algorithms for robotic systems made up from robots that change their position in order to form a given pattern is very important and also challenging due to severe limitations, such as in communication between the robots, hardware constraints, obstacles etc.

The significance of positioning the robots based on some given patterns may be useful for various tasks, such as in bridge building, in forming adjustable buttresses to support collapsing buildings, satellite recovery, or tumor excision [11]. Also, distributed motion planning algorithms for robotic systems are potentially useful in environments that are inhospitable to humans or are hard to directly control and observe (e.g. space, undersea).

In this paper, we consider a system of multiple robots that move on the plane. The robots are anonymous and oblivious, i.e. they cannot be distinguished by a unique `id` or by their appearance, do not have a common x-y coordinate system and are unable to remember past actions. Furthermore, the robots are unable to communicate directly (i.e. via a wireless transmission interface) and

*This work has been partially supported by the IST Programme of the European Union under contract numbers IST-1999-14186 (ALCOM-FT) and IST-2001-33116 (FLAGS).

†Computer Technology Institute, P.O. Box 1122, 26110 Patras, Greece, Emails: {ichatz,nikole}@cti.gr

‡Dept. of Computer Engineering and Informatics, University of Patras, 26500 Patras, Greece. Email: markou@ceid.upatras.gr

can only interact by observing each other's position. Based on this model we study the problem of the robots positioning themselves to form a circle.

Remark that the formation of a circle provides a way for robots to agree on a common origin point and a common unit distance [14]. Such an agreement allows a group of robots to move in formation [9]. In addition, formation of patterns and flocking of a group of mobile robots is also useful for providing communication in ad-hoc mobile networks [3–5].

The main idea is to let each robot execute a simple algorithm and plan its motion adaptively based on the observed movement of other robots, so that the robots as a group will manage to form a given pattern (e.g. a circle).

1.1 Related Work

The problem of forming a circle having a given diameter by identical mobile robots was first discussed by Sugihara and Suzuki [13]. Assuming that the positions of the robots is the only information available to the robots, they proposed a simple heuristic distributed algorithm, which however forms an approximation of a circle (which reminds a Reuleaux triangle). Tanaka [16] later improved their algorithm by avoiding this problem and generating a better approximation of a circle. In an attempt to overcome this problem, Suzuki and Yamasihita [15] propose an algorithm under which the robots eventually reach a configuration where they are arranged at regular intervals on the boundary of a circle. However, to succeed in forming the pattern, the robots must be able to remember all past actions. Lately, Défago and Konogaya [7] designed an algorithm that manages to form a proper circle. In fact, our research builds upon their work and more specifically we work using their system model.

Under a similar model, in which robots have a limited vision, Ando et al. [1] propose an algorithm under which the robots converge to a single point. Flochini et al. [8] study the same problem by assuming that the robots have a common sense of direction (i.e. through a compass) and without considering instantaneous computation and movement.

We here note that there exist other models and problems for the development of motion planning algorithms for robotic systems, e.g. [6, 17]. In that model most of the existing motion planning strategies rely on centralized algorithms to plan and supervise the motion of the system components [6], while recently efficient distributed algorithms have been proposed [17].

1.2 Our Contribution

In this work, based on the system model stated by Défago and Konogaya [7] (which is based on the model of Suzuki and Yamashita [15]), we present a new distributed algorithm that moves a team of anonymous mobile robots in such a way that a (non degenerate) circle is formed. The new algorithm presented here is based on the observation that the Défago - Konogaya algorithm (DK algorithm) [7] is using some very complex computational procedures. Instead of computing the Voronoi diagram based on the locations of the robots and then move towards the smallest enclosing circle (a procedure that is computationally heavy task), our algorithm just moves the robots towards the closest point on the circumference of the smallest enclosing circle.

We prove that our algorithm is correct and that the robots travel shorter distance than when executing the DK algorithm, i.e. the performance of the DK algorithm is an upper bound for the performance our algorithm.

Furthermore we conduct a very detailed comparative simulation experimental study of our algorithm and the DK algorithm, in order to validate the theoretical results and further investigate the performance of the two algorithms. We remark that this is the first work on implementing and simulating the DK algorithm. The experiments show that the execution our algorithm is very simple and takes considerably less time to complete than the DK algorithm. Furthermore, our algorithm seems to be more efficient (both with respect to number of moves and distance travelled) in systems that are made up from a large number of mobile robots.

2 The Model

Let $r_1, r_2, \ldots, r_n$ be a set of extremely small robots, modelled as mobile processors with infinite memory and a sensor to detect the instantaneous position of all robots (i.e. a radar). The robots are able to move with very high precision in an unbounded two dimensional space devoid of any landmark. Each robot r_i uses its own local x-y coordinate system (origin, orientation, distance) and has no particular knowledge of the local coordinate system of other robots, nor of a global coordinate system. It is assumed that initially all robots occupy different positions, although, due to their small size, two or more robots may occupy the same position.

Robots are anonymous in the sense that they are unable to uniquely identify themselves, neither with a unique identification number nor with some external distinctive mark (e.g. color, flag).

All robots execute the same deterministic algorithm, and thus have no way to generate a unique identity for themselves; more generally, we assume that no randomization can be used and in this way, any two independent executions of the algorithm with identical input values always yield the same output.

Time is represented as an infinite sequence of time instants $t_0, t_1, t_2, \ldots$ and at each time instant t, every robot r_i is either *active* or *inactive*. Without loss of generality we assume that at least one robot is active at every time instance. Each time instant during which a robot becomes active, it computes a new position using a given algorithm and moves towards that position. Conversely, when a robot is inactive, it stays still and does not perform any local computation.

We use A_t to denote the set of active robots at t and call the sequence $\mathcal{A} = A_0, A_1, \ldots$ an *activation schedule*. We assume that every robot becomes active at infinite many time instants, but no additional assumptions are made on the timing with which the robots become active. Thus $\mathcal{A}$ needs satisfy only the condition that every robot appears in infinitely many $\mathcal{A}'s$.

Given a robot r_i, $p_i(t)$ denotes its position at time t, according to some global x-y coordinate system (which is not known to the robots) and $p_i(0)$ is its initial position. $P(t) = \{p_i(t) \,||\, 1 \leq i \leq n\}$ denotes the multiset of the position of all robots at time t.

The algorithm that each robot r_i uses is a function ϕ that is executed each time r_i becomes active and determines the new position of r_i, which must be within one distance unit of the previous position, as measured by r_i's own coordinate system. The arguments to ϕ consists of the current position of r_i and the multiset of points containing the observed positions of all robots at the corresponding time instant, expressed in terms of the local coordinate system of r_i. It is assumed that obtaining the information about the system, computing the new position and moving towards it is instantaneous.

Remark that in this paper we consider oblivious algorithms and thus ϕ is not capable of storing any information on past actions or previous observations of the system.

The model we use was proposed by Défago and Konogaya [7], which in turn is based on the model of Suzuki and Yamashita [15].

3 The Problem

In this paper we consider the problem of positioning a set of mobile robots in such a way so that a circle is formed, with finite radius greater than zero. We call such a circle *non degenerate.*

Problem 3.1 (Circle Formation) *Given a group of n robots $r_1, r_2, \ldots, r_n$ with distinct positions, located arbitrarily on the plane, arrange them to eventually form a non degenerate circle.*

We also consider the more difficult problem in which the robots are arranged at regular intervals on the boundary of the circle.

Problem 3.2 (Uniform Circle Formation) *Given a group of n robots $r_1, r_2, \ldots, r_n$ with distinct positions, located arbitrarily on the plane, eventually arrange them to eventually at regular intervals on the boundary of a non degenerate circle.*

A possible way to solve the problem of forming a uniform circle is to form a "simple" circle (i.e. solve prob. 3.1) and then *transform* the robot configuration as follows.

Problem 3.3 (Uniform Transformation) *Given a group of n robots $r_1, r_2, \ldots, r_n$ with distinct positions, located on the boundaries of a non degenerate circle, eventually arrange them at regular intervals on the boundary of the circle.*

We now provide some definitions that we will use in the following sections.

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

Voronoi Diagram. The *Voronoi diagram* of a set of points $P = \{p_1, p_2, \ldots, p_n\}$, denoted by $\mathsf{Voronoi}(P)$, is a subdivision of the plane into n cells, one for each point in P. The cells have the property that a point q in the plane belongs to the Voronoi cell of point p_i, denoted $Vcell_{p_i}(P)$, if and only if, for any other point $p_j \in P, \mathrm{dist}(q, p_i) \leq \mathrm{dist}(p_j, q)$, where $\mathrm{dist}(p, q)$ is the Euclidean distance between two points p and q. In particular, the strict inequality means that points located on the boundary of the Voronoi diagram do not belong to any Voronoi cell. More details on Voronoi diagrams can be found in [2].

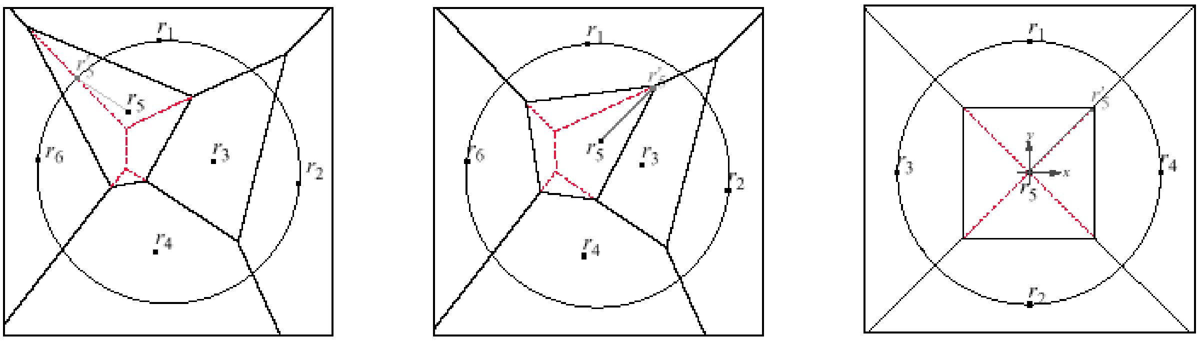
4 A New Circle Formation Algorithm

4.1 The Défago - Konogaya (DK) Algorithm

In this section we briefly describe the Défago-Konogaya circle formation algorithm (which we call the DK algorithm) that is described in [7]. The algorithm relies on two facts: (i) the environment observed by all robots is the same, in spite of the difference in local coordinate system and (ii) the smallest enclosing circle is unique and depends only on the relative positions of the robots. Based on these two facts, the algorithm makes sure that the smallest enclosing circle remains invariant and uses it as a common reference.

The DK algorithm is decomposed in two sub-algorithms. Initially, given an arbitrary configuration in which all robots have distinct positions, the first sub-algorithm (ϕ_{circle}) brings the system towards a configuration in which all robots are located on the boundary of the circle (i.e. solves the Circle Formation problem, prob. 3.1). Then, given that all robots are located on the circumference of a circle, the second sub-algorithm ($\phi_{uniform}$) converges towards a homogeneous distribution of the robots along that circumference, but it does not terminate (i.e. solves the Uniform Transformation problem, prob. 3.3).

Clearly, the combination of the above two sub-algorithms solves the problem of Uniform Circle Formation (prob. 3.2).



(a) The smallest enclosing circle is reachable

(b) The smallest enclosing circle is unreachable

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

Figure 1: Illustration of algorithm ϕ_{circle}, executed by robot r_i (in each case, r_i moves towards r_i')

Circle Formation Algorithm ϕ_{circle}. The main idea of the algorithm is very simple: robots that are already on the boundary of the circle do not move and robots that are in the interior of the circle are made to move towards the boundary of the circle.

When a robot that is located in the interior of the circle is activated, it observes the positions of the other robots and computes the Voronoi diagram based on this observation. Given the boundaries of the smallest enclosing circle and the Voronoi cell where the robot is located, it can find itself in either one of three types of situations:

Case 1: The simplest situation occurs when the circle intersects the Voronoi cell of the robot (see fig. 1a). In this case, the robot selects a point on the intersection of the circle and the Voronoi cell and moves towards it.

Case 2: When the Voronoi cell of the robot does not intersect with the circle (see fig. 1b), the robot selects the point in its Voronoi cell which is nearest to the boundary of the circle (or farthest from its center).

Case 3: The third situation arises when, due to symmetry, there exist several points (see fig. 1c). In this case, all solutions being the same, one is selected arbitrarily. This is for instance done by keeping the solution with the highest x-coordinate (and then y-coordinate) according to the local coordinate system of the robot.

The authors prove that their algorithm is correct (*Theorem 2*, [7]) and solves the problem of Circle Formation (prob. 3.1).

Uniform Transformation Algorithm $\phi_{uniform}$. Given that all robots are located on the circumference of a circle, the algorithm $\phi_{uniform}$ converges toward a homogeneous distribution of robots, but does not terminate deterministically. The algorithm $\phi_{uniform}$ works as follows: when a robot r_i becomes active, it considers its two direct neighbors *prev*(r_i) and *next*(r_i) and computes the midpoint between them and moves halfway towards it.

The reason for moving halfway toward the midpoint rather than toward the midpoint itself is to prevent situations where the system oscillates endlessly between two different configurations when robots are perfectly synchronized. The system would get stuck into an infinite cycle and hence be unable to progress toward an acceptable solution.

The authors prove that algorithm $\phi_{uniform}$ converges toward a configuration wherein all robots are arranged at regular intervals on the boundary of a circle (*Theorem 3*, [7]) and thus solves the problem of Uniform Transformation (prob. 3.3).

4.2 Our Algorithm

In this section we present a new distributed algorithm (which we call direct) that moves a set of oblivious mobile robots in such a way that a (non degenerate) circle is formed . The new algorithm is motivated by the observation that the DK algorithm is using some very complex (i.e. computationally intensive) procedures. Indeed, computing the Voronoi diagram based on the locations of the robots and then moving towards the smallest enclosing circle, involves computationally complex procedures. Instead, our algorithm is much simple since it just moves the robots towards the closest point on the circumference of the smallest enclosing circle.

In more details, under algorithm direct a robot can find itself in either one of the following two situations:

Case 1: Robot r is located on the boundary of $SEC(P)$. In this case, r stays still.

Case 2: Robot r is located inside $SEC(P)$. In this case, r selects the closest point of the boundary of $SEC(P)$ to its current position and moves towards this point.

In fig. 2, we provide a pseudo-code description.

function $\phi'_{circle}(P, p_i)$

```
begin
   if (p_i ∈ SEC(P) == true)
     begin
       stay still
     end
   else
     begin
       target := point on boundary of SEC(P) closest to p_i
       move toward target
     end
end
```

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

4.2.1 Correctness

In order to prove the correctness of algorithm direct we work in a similar way as in [7]. We first prove that when all robots are positioned on the boundary of the smallest enclosing circle, the algorithm terminates (Lemma 4.1). We then show that the robots that are not positioned on the boundary of $SEC(P)$ will always move towards the boundary (and not further away from $SEC(P)$, Lemma 4.2). Finally we show that a finite period of time is needed for those robots that are positioned inside $SEC(P)$ to move towards the boundary of $SEC(P)$ (Lemma 4.4).

Lemma 4.1 Under Algorithm direct, all configurations in which the smallest enclosing circle passes through all robots are stable.

Proof: In such configurations, it is $p_i \in SEC(P), \forall i \in \{1, 2, \ldots, n\}$. Thus condition 2 in the pseudo-code description is satisfied for all robots, thus every robot stays still (line 4) and the total configuration is stable. ■

Lemma 4.2 No robot moves to a position further away from $SEC(P)$ (than its current position).

Proof: Under algorithm direct, robots move only at lines 4 and 9. Let us consider each case for some arbitrary robot r.

1. At line 4, r stays still, so it obviously does not move towards the center.
2. At line 9, r moves from the interior of the circle toward a point located on the boundary of the circle. So, r actually progresses *away* from the center.

Thus, the robot r is unable to move further away (i.e. towards the circle center) from the boundary of the smallest enclosing circle in any of the two cases.

■

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

Proof: Under algorithm direct the motion of the robots that are located in the interior of the smallest enclosing circle, is not blocked by other robots, since each robot's movement in our algorithm is independent of the positions of other robots. Therefore, at any given time instance t, robot r_i that is active and is located in the interior of the smallest enclosing circle, will move towards the boundary of the circle.

Thus, if a set of robots is located in the interior of $SEC(P)$ and at least one robot is active, there exists at least one robot (an active one) in the interior of $SEC(P)$ that can progress a non null distance towards $SEC(P)$. This hence proves the lemma. ■

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

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

Theorem 4.5 *Algorithm* direct *solves the problem of Circle Formation (prob. 3.1).*

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

Corollary 4.6 The combination of functions ϕ'_{circle} and $\phi_{uniform}$ solves the problem of Uniform Circle Formation (prob. 3.2).

4.2.2 Analytic Considerations

We now provide a comparison of the performance of our algorithm to that of the DK algorithm. More specifically, we show that the distance covered by the robots when executing algorithm direct is at most the distance covered when executing algorithm DK. This is the initial intuition that led us to the design of the new algorithm.

Theorem 4.7 A robot r that executes algorithm direct covers distance less or equal to the distance covered by r when executing function ϕ_{circle} of algorithm DK, given that the initial conditions of the system are identical. In other words, the distance covered by r when executing ϕ_{circle} is an *upper bound* for the distance covered by r when executing algorithm direct.

Proof: Let r be a robot positioned at point A in the plane which is inside the boundaries of $SEC(P)$ (see fig. 3a) and r is executing algorithm direct. When r is activated, it will move on the line that connects the center of $SEC(P)$ (point c) and A. Regardless of the number of steps that will be required in order to reach the boundary of the circle, r will keep moving along the line cA. Let B the point where cA intersects $SEC(P)$. Therefore the distance covered by r is equal to $|AB|$.

Now let assume that r is executing function ϕ_{circle} of algorithm DK. When r is activated, it will start moving away from A towards a point B' located on the boundary of $SEC(P)$, i.e. not necessarily on the line AB.

Let C be a circle centered on A having radius equal to $|AB|$. We observe that any point B' (other than B) that lies on the boundary of $SEC(P)$ is positioned outside the boundary of C and thus, based on the definition of the circle, further from the center of C (i.e. A). Therefore $|AB| < |AB'|$.

In the case when r is moving towards B' after first passing through a point D outside the C circle, its route will clearly greater length than AB' (by the triangle inequality) and thus greater than AB. For instance, considering the example of fig. 3b, it holds that: $|AD| + |DB'| > |AB'| > |AB|$.

Thus, if r follows any other path than AB in order to reach the boundary of $SEC(P)$, it will cover grater distance, which proves the theorem. ■

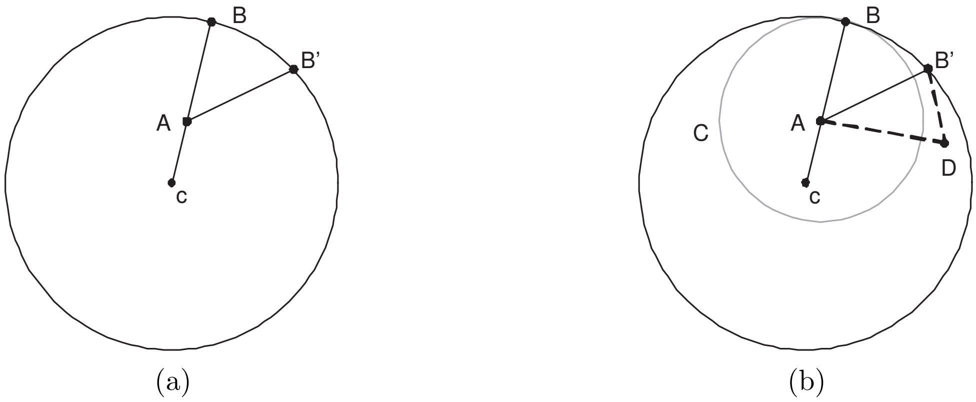


Figure 3: Distance comparison of next position under the two algorithms

5 Experimental Evaluation

In order to evaluate the performance of the two protocols and further investigate their behavior we carried a comparative experimental study. All of our implementations follow closely the protocols described above. They have been implemented as C++ classes using several advanced two-dimensional geometry data types and algorithms of LEDA [10].

The simulation environment that we developed is based on the model presented in section 2 and allows to perform an experiment for given dimensions of the Euclidean space, a fixed number of mobile robots and a specified number of repetitions. After the execution of the simulation, the environment stores the results on files so that the measurements can be represented in a graphical way.

The experiments were conducted on a Linux box (Mandrake 8.2, Pentium III at $933Mhz$, with $512MB$ memory at $133Mhz$) using g++ compiler $ver.$2.95.3 and LEDA library $ver.$4.1.

5.1 Efficiency Measures

In order to evaluate the performance of the above protocols we define a set of efficiency measures. Recall that there are two different problems that we address in this paper, the circle formation (prob. 3.1) and the uniform circle formation (prob. 3.2). Thus the measurement of the efficiency of each protocol is considered for each problem separately.

We start by defining the average number of steps performed by each robot. That is, we measure the total number of steps performed by robot r_i when executing function ϕ and then calculate the average number of steps dividing by the number of robots. More formally this is defined as follows:

Definition 5.1 Let M^i_ϕ be the *total number of steps* performed by robot r_i when executing function ϕ. Let $\text{avg}M_\phi = \frac{\sum_{i=1}^n M^i_\phi}{n}$ be the *average number of steps* required for function ϕ to form a given pattern, where n is the number of robots in the system.

Remark that although the efficiency measure defined above provides a way to compare the performance of the two protocols, we cannot use the total number of steps for measuring the time complexity of a formation algorithm since a robot may remain inactive for an unpredictable period of time. An alternative measure, also proposed in [15], is the total distance that a robot must move to form a given pattern.

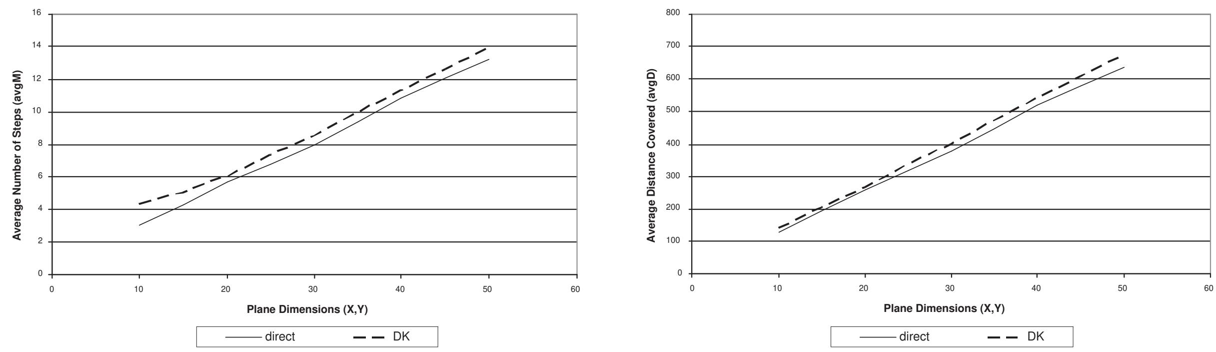
Definition 5.2 Let D^i_ϕ be the *total distance* covered by robot r_i when executing function ϕ. Let $\text{avg}D_\phi = \frac{\sum_{i=1}^{n} D^i_\phi}{n}$ be the *average distance* required for function ϕ to form a given pattern, where n is the number of robots in the system.

Finally, another important efficiency measure is the total time of execution required by function ϕ to form a given pattern. The computational complexity is a very important efficiency measure since formation algorithms are executed in real time and the computational power of the mobile modules can be a limiting factor. Although the function ϕ is executed by each robot in a distributed fashion, we consider the overall time required by the robots until the given pattern is formed. Remark that when calculating the execution of function ϕ we assume that the motion of the robots takes negligible time (i.e. the motion of the robots is instantaneous).

Definition 5.3 Let T^i_ϕ be the *execution time* required by robot r_i to execute function ϕ. Let $\text{tot}T = \sum_{i=1}^{n} T^i_\phi$ be the *total execution time* required for function ϕ to form a given pattern, where n is the number of robots in the system.

5.2 Experimental Results and Discussion

We start our experimentation by considering the effect of the dimensions of the Euclidean space where the mobile robots move on the performance of the two algorithms. More specifically we use a fixed number of mobile robots (100) on an Euclidian plane of dimensions $X \in [10, 50]$ and $X = Y$. The robots are positioned using a random uniform distribution. Each experiment is repeated for at least 100 times in order to get good average results.



(a) Average Number of Steps (avg$M$) (b) Average Distance Covered (avg$D$)

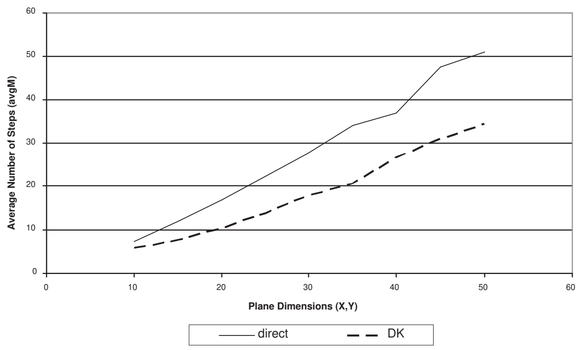
Figure 4: Circle Formation: Evaluating the performance for various Plane Dimensions

Figure 4 depicts the average number of steps and average distance travelled under the two algorithms for the *circle formation problem* (i.e. the first phase). We observe that the performance of the two algorithms is more or less linear in the dimensions of the plane. Actually, it seems that the gradient of the lines is similar for both algorithms. It is evident that algorithm direct manages to form a circle requiring slightly fewer number of steps than the DK algorithm and the robots move in a way such that less distance is covered. This result verifies our theoretical upper bound provided in sec. 4.2.2.

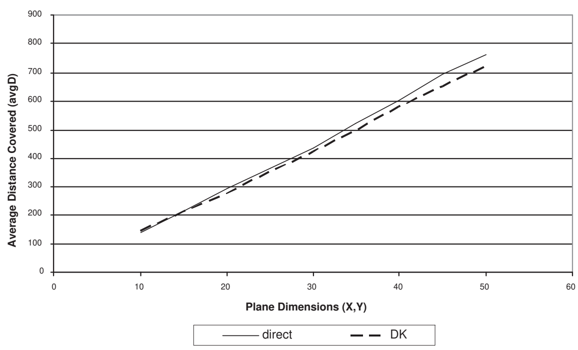
The results of the second phase of the algorithm, i.e. for the *uniform transformation problem*, are shown in fig. 5 for the same efficiency measures. In this phase, the DK algorithm is performing substantially better regarding the average number of steps while the robots cover slightly less distance than in algorithm direct. Again we observe that the performance of the two algorithms is more or less linear in the dimensions of the plane.

In fig. 6 we get the combination of the results for the first and second phase, i.e. when considering the *uniform circle formation problem.* First, we observe that the average number of steps performed by the robots (see fig. 6a) in the second phase ($\phi_{uniform}$) is dominating the overall performance of the two algorithms. Thus, the DK algorithm manages to form a uniform circle with a much smaller average number of steps per robot. However, regardless of the number of steps, under both algorithms the robots seem to cover similar distances (see fig. 6b).

Figure 7 depicts the total time required to execute each algorithm for both phases. Clearly, algorithm direct executes in significantly less time and actually the graph suggests that it takes about 60 times less than the DK algorithm. It is evident that algorithm direct is very simple to execute making the total execution time almost independent from the plane dimensions. On the other hand the execution time of the DK algorithm seems to be linear to the plane dimensions.

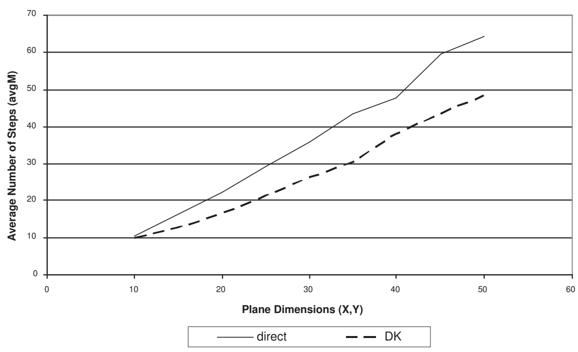


(a) Average Number of Steps (avg$M$)

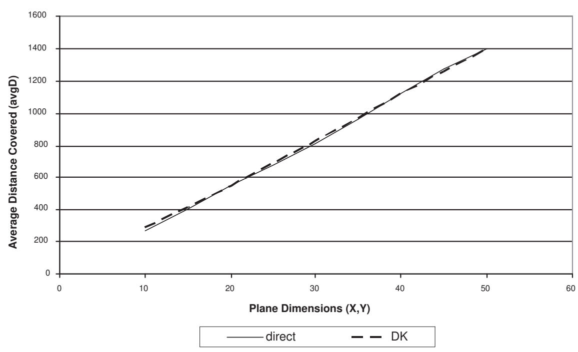


(b) Average Distance Covered (avg$D$)

Figure 5: Uniform Transformation: Evaluating the performance for various Plane Dimensions



(a) Average Number of Steps (avg$M$)



(b) Average Distance Covered (avg$D$)

Figure 6: Uniform Circle Formation: Evaluating the performance for various Plane Dimensions

In the second set of experiments we investigate the performance of the two algorithms as the number of mobile robots increases while keeping the dimensions of the Euclidean plane fixed; that is, we investigate the effect of the density of robots on the performance of the two algorithms. In particular we keep the Euclidean plane size fixed ($X = Y = 20$) and vary the number of mobile

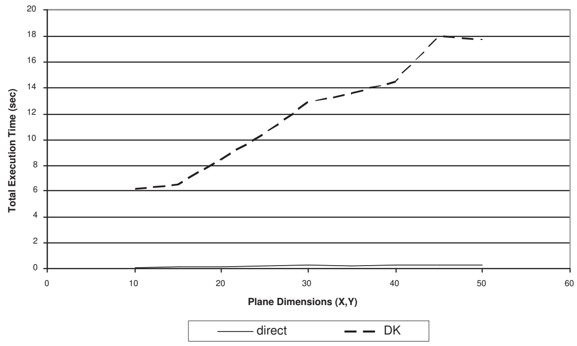
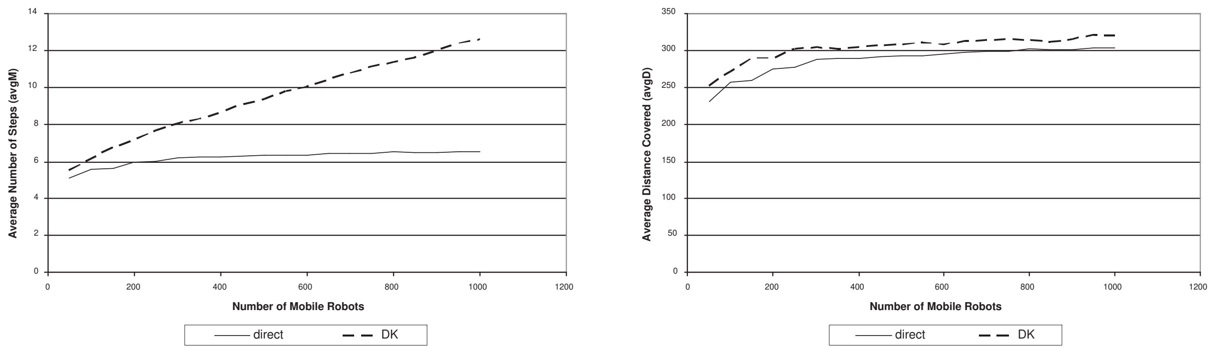


Figure 7: Uniform Circle Formation: Total Execution Time (avg$T$) vs. Plane Dimensions

robots $m \in [50, 1000]$. The robots are positioned using a random uniform distribution. Each experiment is repeated for at least 100 times in order to get good average results.

We start by considering the first phase of the two algorithms (i.e. the *circle formation problem*). Figure 8a depicts the average number of steps as the density of robots increases. The graph shows that the performance of the DK algorithm, in terms of average number of steps, is linear in the total number of mobile robots. On the other hand, algorithm direct seems to be unaffected by this system parameter, especially when $m \geq 200$. This threshold behavior of the two algorithms is also observed in fig. 8b. For low densities of robots, the average distance increases slowly with m until a certain number of robots is reached ($m = 200$) when no further increase is observed. Again, we observe that algorithm direct manages to form a circle requiring the robots to move in a way such that less distance is covered. This also verifies the theoretical upper bound provided in sec. 4.2.2.

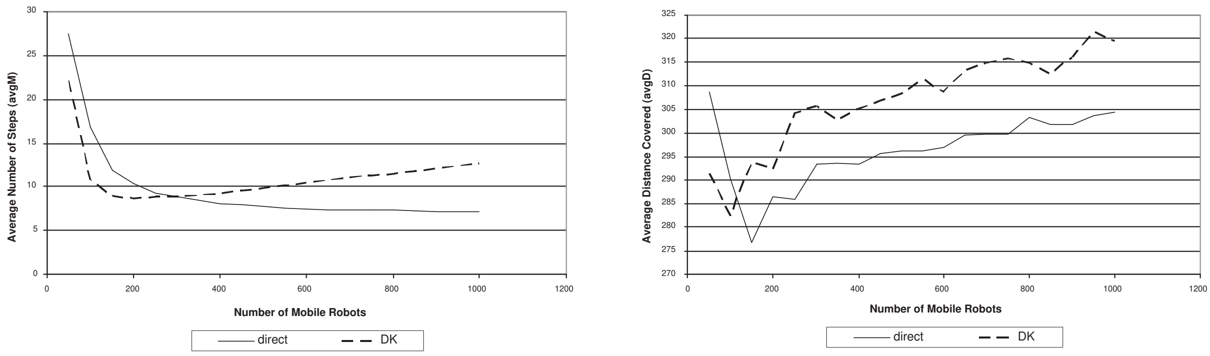


(a) Average Number of Steps (avg$M$) (b) Average Distance Covered (avg$D$)

Figure 8: Circle Formation: Evaluating the performance as the number of Mobile Robots increases

Regarding the second phase, in figure 9a we observe that both algorithms require a high number of steps, which, interestingly, decreases as the density of robots increases. However, for the DK algorithm, when the number of robots crosses the value $m = 200$, the average number of steps starts to increase linearly to the number of robots that make up the system. On the other hand, when $m \geq 200$, the performance of algorithm direct seems to remain unaffected by the density of the robots. As a result, when $m = 300$ the two algorithms achieve the same performance, while for $m \geq 300$ algorithm direct outperforms the DK algorithm.

The average distance covered by the robots under the two different algorithms, as the density of the system increases is shown in figure 9b. In this graph we observe that the two algorithms have a different behavior than that in the first phase. More specifically, initially the performance of the algorithms drops very fast as the density of the system increases until a certain value is reached ($m = 200$) when the performance starts to increase linearly with m. Furthermore, taking into account the statistical error of our experiments, fig. 9b suggests that when the robots execute algorithm direct they travel about 5% less distance than when executing algorithm DK.



(a) Average Number of Steps (avg$M$) (b) Average Distance Covered (avg$D$)

Figure 9: Uniform Transformation: Evaluating the performance as the number of Mobile Robots increases

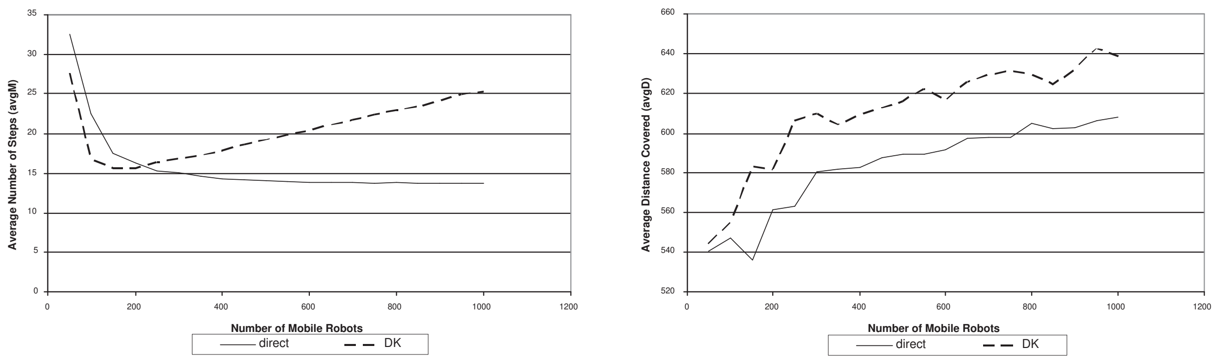
When considering the *uniform circle formation* problem (i.e. for both phases of the algorithms) it is clear that the overall behavior of the two algorithm is dominated by the second phase. It seems that algorithm direct requires a constant number of steps when $m \geq 300$ while the DK algorithm's performance is linear to the number of mobile robots that make up the system. This observation is depicted in fig. 10a. As for the average distance covered, again, the behavior of the algorithms is dominated by the second phase. Fig. 10b follows the same pattern as in fig. 9b.

Finally, in fig. 11 depicts the overall time required by the robots to execute each algorithm. In this figure we clearly see that the DK algorithm has a very poor performance as the density of the robots in the system increases. On the other hand, the performance of algorithm direct is independent from the number of robots and is clearly more suitable for systems comprised of very large number of robots.

6 Closing Remarks

We presented a new algorithm for the problem of circle formation for systems made up from anonymous mobile robots that cannot remember past actions. We provided a proof of correctness (i.e. the algorithm always succeed in positioning the robots uniformly on the boundary of a circle) and provided an upper bound on its performance.

Furthermore, we conducted an extensive set of experiments in order to validate the theoretical results and further investigate the performance of our algorithm when compared to the Défago - Konogaya (DK) algorithm [7]. The experiments show that the execution our algorithm is very simple and takes considerably less time to complete than the DK algorithm. Furthermore, in systems that are made up from a large number of mobile robots, our algorithm is more efficient than DK (both with respect to number of moves and distance travelled by the robots).



(a) Average Number of Steps (avg$M$) (b) Average Distance Covered (avg$D$)

Figure 10: Uniform Circle Formation: Evaluating the performance as the number of Mobile Robots increases

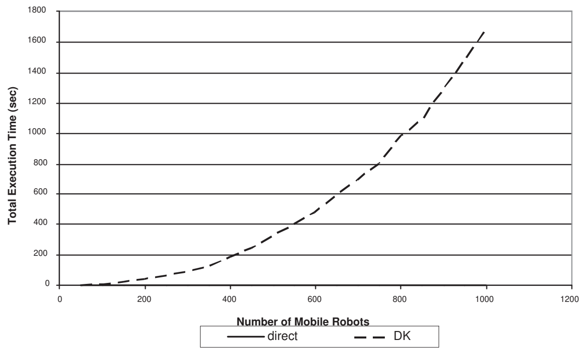


Figure 11: Uniform Circle Formation: Total Execution Time (avg$T$) as the number of Mobile Robots increases

References

[1] H. Ando, I. Suzuki, and M. Yamashita, *Distributed memoryless point convergence algorithm for mobile robots with limited visibility*, Trans. on Robotics and Automation **15** (1999), no. 5, 818–828.

[2] F. Aurenhammer, *Voronoi diagrams - a survey of a fundamental geometric data structure*, Comput. Surv. **23** (1991), no. 3, 345–405.

[3] I. Chatzigiannakis, *Design and analysis of distributed algorithms for basic communication in ad-hoc mobile networks*, Ph.D. dissertation, Dept. of Computer Engineering and Informatics, University of Patras, Greece, May 2003.

[4] I. Chatzigiannakis, S. Nikoletseas, and P. Spirakis, *Smart dust protocols for local detection and propagation*, 2nd International Annual Workshop on Principles of Mobile Computing (POMC 2002), ACM, 2002, pp. 9–16.

[5] I. Chatzigiannakis, S. Nikoletseas, and P. Spirakis, *Distributed communication algorithms for ad hoc mobile networks*, Journal of Parallel and Distributed Computing **63** (2003), no. 1, 58–74.

[6] G. Chirikjian, *Kinematics of a metamorphic robotic system*, Conference on Robotics and Automation, IEEE, 1994, pp. 449–455.

[7] X. Défago and A. Konagaya, *Circle formation for oblivious anonymous mobile robots with no common sense of orientation*, 2nd International Annual Workshop on Principles of Mobile Computing (POMC 2002), ACM, 2002.

[8] P. Flocchini, G. Prencipe, N. Santoro, and P. Widmayer, *Gathering of asynchronous oblivious robots with limited visibility*, 10th International Symposium on Algorithms and Computation (ISAAC 1999), Springer-Verlag, 1999, Lecture Notes in Computer Science, LNCS 1741, pp. 93–102.

[9] V. Gervasi and G. Prencipe, *Flocking by a set of autonomous mobile robots*, Tech. report, Dipartmento di Informatica, Università di Pisa, October 2001, TR-01-24.

[10] K. Mehlhorn and S. Näher, *LEDA: A platform for combinatorial and geometric computing*, Cambridge University Press, 1999.

[11] A. Pamecha, I. Ebert-Uphoff, and G. Chirikjian, *Useful metrics for modular robot motion planning*, Transactions on Robotics and Automation **13** (1997), no. 4, 531–545.

[12] S. Skyum, *A simple algorithm for computing the smallest enclosing circle*, Information Processing Letters **37** (1991), no. 3, 121–125.

[13] K. Sugihara and I. Suzuki, *Distributed motion coordination of multiple mobile robots*, International Symposium on Intelligence Control, IEEE, 1990, pp. 138,143.

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

[15] I. Suzuki and M. Yamashita, *Distributed anonymous mobile robots: Formation of geometric patterns*, J. Computer Science **28** (1999), no. 4, 1347–1363.

[16] O. Tanaka, *Forming a circle of distributed anonymous mobile robots*, Bachelor thesis, Dept. of Electrical Engineering, Hiroshima University, 1992.

[17] J.E. Walter, J.L. Welch, and N.M. Amato, *Distributed reconfiguration of metamorphic robot chains*, 19th ACM Annual Symposium on Principles of Distributed Computing (PODC 2000), ACM, 2000, pp. 171–180.