# Biangular Circle Formation by Asynchronous Mobile Robots*

Branislav Katreniak

Faculty of Mathematics, Physics and Informatics,
Comenius University, Bratislava
katreniak@dcs.fmph.uniba.sk

**Abstract.** Consider a community of simple autonomous robots freely moving in the plane. The robots are decentralized, asynchronous, deterministic without the common coordination system, identities, direct communication, memory of the past, but with the ability to sense the positions of the other robots. We study the problem of forming an absolutely symmetric formation – regular circle. Unlike the existing algorithms for similar problems that have supposed a stronger model and guaranteed only convergence to a final formation, we are interested in solving the problem for fully asynchronous model. Unfortunately, the problem in general is very hard under these circumstances and seems to be unsolvable. We present an algorithm that solves an intermediate problem, the biangular circle formation, deterministically in finite time.

## 1 Introduction

We consider a distributed system consisting of very weak autonomous mobile robots. The robots are anonymous, have no common knowledge, no common sense of the direction (e.g. compass), no central coordination and no means of direct communication. The study of such a weak system is motivated by the question of the minimal complexity of the mobile devices needed to perform non-trivial tasks. While a number of results deal with heuristics and practical applications, deterministic worst-case solutions are rare. We use the asynchronous model introduced in [4] where the task of arbitrary pattern formation is addressed. It was shown that with a common sense of the direction the arbitrary pattern formation problem is solvable. Moreover, without the common sense of the direction there are patterns which are not formable. In [2], [3] it was shown that the gathering problem is solvable with multiplicity detection and in limited visibility model with compass. We use the basic model without any extensions and study the problem of the pattern formation for the particular pattern – biangular circle. The only similar problem (convergence to the regular circle pattern) was addressed [1] only in a semi-synchronous model.

* Supported in part by APVT 20-018902.

A. Pelc and M. Raynal (Eds.): SIROCCO 2005, LNCS 3499, pp. 185–199, 2005.

Consider the life-cycle of a robot: Initially it is in a waiting state, wakes up asynchronously, observes the other robots' positions, computes a point in the plane, moves toward this point (but may not reach it) and becomes waiting again. Each step takes an unpredictable amount of time. Robots are oblivious, i.e. the only input for their computation is the currently observed set of the other robots' positions.

First, we only require the robots to move onto the boundary of a *circle*. This problem is easy, every robot just takes the shortest way to the smallest enclosing circle ($SEC$) of all robots. The $SEC$ will never move and all robots will reach $SEC$ in a finite time.

In section 4 we show how the robots can reach the boundary of a circle at *distinct positions*, provided they are located at distinct positions at the start. Only the robots closest to $SEC$ will take the shortest way to $SEC$, others will wait until it is secure for them to do a *side-step* – an atomic move along the concentric circle.

Finally, we would like the robots to form[1] a *regular circle*. This problem is very hard and seems to be unsolvable in general. In section 5 we show an intermediate result: Robots try to form a regular circle, but they do not achieve it in all cases. In general they form only a bit less symmetric pattern, the *biangular circle*. The main idea of the algorithm is the synchronization of the asynchronous robots. We present a restricted pseudo-synchronous model with robots moving along the boundary of a fixed circle and show its emulation in our asynchronous model. Then this pseudo-synchronous circle model is used to transform the circle pattern into the biangular circle pattern.

## 2 Model

We use the model introduced in [4]. Each robot is viewed as a point in a plane equipped with sensors. It can observe the set of all points which are occupied by at least one other robot. Note that the robot only knows whether there are other robots at a specific point, but it has no knowledge about their number (i.e. it cannot tell how many robots are at a given location). The *local view* of each robot consists of a unit of length, an origin (w.l.o.g. the position of the robot in its current observation), an orientation of angles and the coordinates of the observed points. No kind of agreement on the unit of length, the origin or the orientation of angles is assumed among the robots.

A robot is initially in a *waiting* state (*Wait*). Asynchronously and independently from the other robots, it *observes* the environment (*Look*) by activating its sensors. The sensors return a snapshot of the world, i.e. the set of all points occupied by at least one other robot, with respect to the local coordinate system. Then, based only on its local view of the world, the robot *calculates* its destination point (*Compute*) according to its deterministic algorithm (the same

[1] We again suppose that robots are at the start located at distinct positions.

for all robots). After the computation the robot *moves* towards its destination point (*Move*); if the destination point is the current location, the robot stays on its place. A move may stop before the robot reaches its destination (e.g. because of limits to the robot's motion energy). The robot then returns to the waiting state. The sequence *Wait* – *Look* – *Compute* – *Move* forms a *cycle* of a robot.

The robots are *fully asynchronous*, the amount of time spent in each phase of a cycle is finite but otherwise unpredictable. In particular, the robots do not have a common notion of time. As a result, robots can be seen by the other robots while moving, and thus computations can be made based on obsolete observations. The robots are *oblivious*, meaning that they do not remember any previous observations nor computations performed in any previous cycles. The robots are *anonymous*, meaning that they are indistinguishable by their appearance, and they do not have any kind of identifiers that can be used during the computation. Finally, the robots have *no means of direct communication*: any communication occurs in a totally implicit manner, by observing the other robots' positions.

There are two limiting assumptions concerning *infinity*:

**Assumption 1.** *The amount of time required by a robot to complete a cycle is finite and bounded from above be a fixed constant.*

**Assumption 2.** *The distance traveled by a robot in a cycle is bounded from below be a fixed $\varepsilon$ (or the robot gets to the destination point).*

As no other assumptions on space exist, the distance traveled by a robot in a cycle is unpredictable.

## 3 Notations

In general, $r$ denotes the robot itself or its position in the plane. The *configuration* of robots $R$ at a given time is the set of positions in the plane occupied by the robots; $n$ is the number of robots in $R$.

Given two distinct points $a$ and $b$ in the plane, $[a, b)$ denotes the half-line that starts in $a$ and passes through $b$; $\texttt{flip}([a, b))$ denotes $[a, b)$ rotated by 180° about $a$; $[a, b]$ denotes the line segment between $a$ and $b$; $|a, b|$ the Euclidean distance between the points $a$ and $b$. Given two half-lines $[c, a)$ and $[c, b)$, we denote by $\sphericalangle(a, c, b)$ the clockwise angle centered in $c$ from $[c, a)$ to $[c, b)$; by $|\sphericalangle(a, c, b)|$ the size of this angle; by $\texttt{axis}(\sphericalangle(a, c, b))$ the axis of this angle.

Given a circle $C$ with center $c$ and radius $rad$, we say that robot $r$ *is on* $C$ ($r \in C$) if $|r, c| = rad$ (i.e. $r$ is on the circumference of $C$); $r$ *is inside* $C$ if $|r, c| < rad$; radius($r$) in $C$ is the line segment $[c, q]$, where $q$ is the intersection between the circumference of $C$ and $[c, r)$. We say that two robots $r$ and $r'$ are on the same radius in $C$, if $r' \in$ radius($r$).

Points in the plane form a *biangular situation* (see Figure 1.a) if there exists a point $c$ (the *center of biangularity*), a polar ordering of the points around $c$,

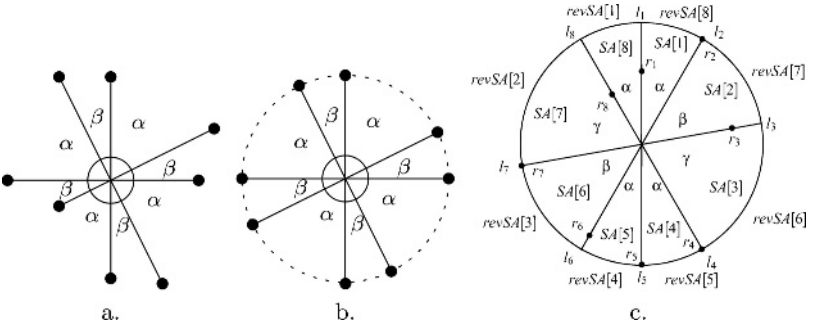


**Fig. 1.** (a) Biangular situation, (b) Biangular circle; (c) String of angles $\mathrm{SA}(r_1)$

and two nonzero angles $\alpha$, $\beta$ such that each two adjacent points form with $c$ angle $\alpha$ or $\beta$ and the angles alternate.

Points in the plane form a *biangular circle* (see figure 1.b) if they are simultaneously in the biangular situation and on the boundary of a circle with the center of the biangularity equal to the center of the circle.

Given a set of points $R$ in the plane, the *smallest enclosing circle* of the points is the circle with minimal radius such that all points from $R$ are inside or on the circle. We denote the circle by $SEC$ and its center by $c$, the set $R$ is always unambiguous from the context. The smallest enclosing circle of a set of $n$ points is unique and can be computed in polynomial time[2].

Given a set of points $R$ in the plane, their $SEC$ and $c$, the *successor* $\mathrm{Succ}(r)$ of any point $r$ is

- either the point $r_i \neq r, r_i \in [c, r]$ with minimal $|r, r_i|$ if such a point exists
- or the point $r_i \neq r$ such that there is no other point inside the angle $\sphericalangle(r, c, r_i)$ and there is no other point on the radius$(r_i)$ further from $c$

We will denote by $\mathrm{Succ}^{(i)}(r)$ the $i$-th power of $\mathrm{Succ}(r)$.

Given a set of points $R$ in the plane, the *string of angles* $\mathrm{SA}(r)$ of any point $r$ is the sequence $\{|\sphericalangle(\mathrm{Succ}^{(i-1)}(r), c, \mathrm{Succ}^{(i)}(r))|;\ 1 \leq i \leq n\}$ (see Figure 1c). The *reverse string of angles* $\mathrm{revSA}(r)$ is defined in the same way as string of angles, but all angles are counterclockwise oriented[3] (i.e. $\mathrm{revSA}(r)$ is the reverse of $\mathrm{SA}(r)$). Instead of $\mathrm{SA}(r)$ we will write only SA if we do not consider a specific point $r$ or when the point $r$ is unambiguous from the context. Make a set of all SA and revSA starting in all points and keep only the lexicographically smallest ones. The resulting subset are the *the lexicographically first strings of angles* (LFSA). We will call the size of LFSA the *degree of symmetry* and denote by $k$.

[2] E.g. take every pair and triple of robots and verify the circle defined by them.

[3] The robots do not have the common sense of clockwise direction, but every individual robot can locally distinguish between a clockwise and counterclockwise orientation.

## 4 Circle Formation

*Problem 1 (Circle formation).*
Given is a group of $n$ robots on distinct positions in the plane, arrange them on the boundary of a circle on distinct positions.

We will keep the smallest enclosing circle $SEC$ invariant and use a polar coordinate system based on it. The center of the polar coordinate system is $c$, the unit of distance *rad*. Angle 0 and clockwise direction are defined only locally for every robot. Angle 0 is either the radius the robot stays on if $c \neq r$, or the robot's local angle 0 otherwise. The clockwise direction is the robot's local one. We will denote $(a, \alpha)_P$ the point at the distance $a$ from $c$ at the angle $\alpha$ (e.g. $(1, 0)_P$ is the intersection of $[c, r)$ and $SEC$).

Algorithm `Circle_formation` implements the solution: Any robot on $SEC$ stays on its place. All others try to get on $SEC$ along their radii. If more then one robot is on the same radius, only the closest one to $SEC$ will move towards $SEC$. The others have to wait until they become one of the closest one to $c$ and then do a *side-step*, an atomic move along the concentric circle, to one third of the angle to the next robot. Side-steps are allowed at most in the distance $\frac{1}{2}$ from the center[4]. If a robot wants to do a side-step further than $\frac{1}{2}$ from $c$, it will move along its radius to this distance first. If the robot is staying on $c$, it will move anywhere close enough to $c$ to be the nearest to $c$ also in its next cycle.

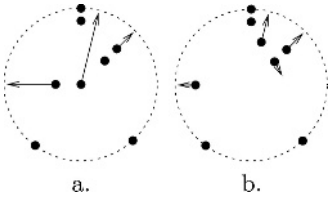


**Fig. 2.** Circle formation algorithm

### 4.1 Correctness

**Lemma 1.** *The smallest enclosing circle SEC is invariant.*

*Proof.* No robot ever leaves $SEC$ and no robot ever moves behind the boundary of $SEC$.

**Lemma 2.** *If a robot did a side-step, there are no other robots on its radius in its new life-cycle and it will visit no occupied radius during its side-step.*

*Proof.* If a robot does a side-step, other robots on the same radius do not leave it (they cannot be the closest to $c$) until it finishes the side-step. Robot will occupy

[4] We will use it in chapter 5.

```
CIRCLE_FORMATION(R, r)
 1  d ← Min(|r_i, c|; r_i ∈ R); e ← Min(|r_i, c|; r_i ∈ R, r_i ≠ r)
 2  if (∄r_i ∈ R; r_i ≠ r; r_i ∈ [r, (1, 0)_P])
 3     then return (1, 0)_P
 4  if |r, c| > d
 5     then return r
 6  if r = c
 7     then return (e/2, 0)_P
 8  if |r, c| > 1/2
 9     then return (1/2, 0)_P
10  return (d, |◁(r,c,Succ(r))|/3)_P
```

new radius in its next cycle (Assumption 2). As the robot's new radius is max. in one third to the nearest old occupied radius and so it is for each other robot, no occupied radius was visited and no other robot stays on the destination radius.

**Lemma 3.** *If a robot chooses the direct way to SEC, it will choose it in all its next life-cycles until it gets there.*

*Proof.* If a robot $r$ chose the direct way to $SEC$, there is no other robot on the same radius closer to $SEC$. Other robots on the same radius cannot cross its way and so is it for all others (Lemma 2).

**Lemma 4.** *Every robot will reach SEC in finite time.*

*Proof.* Consider a robot nearest to the $c$. It will optionally move to the distance $\frac{1}{2}$ and do a side-step. But then it always chooses direct way to $SEC$ and gets there in finite time (Assumptions 1, 2). Eventually, every robot becomes the nearest to $c$.

## 5 Biangular Circle Formation

*Problem 2 (Biangular circle formation).*
Given is a group of $n$ robots on distinct positions, arrange them in a biangular circle pattern.

For at most two robots, the problem is trivially solved – robots can simply stay at their positions. Due to a lack of space the special algorithms for three and four robots are not included. We solve the problem for at least five robots here.

We will keep the smallest enclosing circle $SEC$ invariant and base the polar coordinate system on it in the same way as we did in the circle formation problem. We only override the clockwise direction in some special cases.

### 5.1 Pseudo-synchronous Circle Model

Imagine the pseudo-synchronous circle model (PSC model) where robots move along the boundary of a fixed circle $C$. Each step takes finite time and has three phases:

1. Robots synchronously observe the configuration.
2. Robots compute their destination points.
3. Robots asynchronously move to their destinations. Every robot may stop before it reaches its destination (even not move at all). But at least one moving robot (if such a robot exists) must pass a distance greater than some fixed $\varepsilon$ or get to the destination.

We call a robot *elected*, if its position at the start of a step is not equal to its computed destination point (i.e. it wants to move). We will denote by $\mathcal{M}(\cdot)$ the set of elected robots.

The algorithm used in computation phase must meet the following rules:

**Assumption 3.** *Every robot is able to compute* $\mathcal{M}(\cdot)$.

**Assumption 4.** *It is guaranteed that all robots are at distinct positions during the movement phase.*

**Assumption 5.** *The degree of symmetry cannot be higher during the robots' movement than it was in the observed configuration. If at least one moving robot did not move, the degree of symmetry must be lower.*

### 5.2 Emulation of PSC Model in Asynchronous Model

We define the circle $C$ being equal to $SEC$ ($SEC$ must be invariant) and any robot's position as the intersection of its radius and $SEC$ (as the projection on $SEC$).

Algorithm `Emulate_PSC` implements the emulation. Having observed the configuration, robot calculates:

- The set of elected robots $\mathcal{M}(\cdot)$ (Assumption 3).
- Destination angle $\mathcal{A}(\cdot)$ in PSC model[5].
- Distance $\mathcal{D}(\cdot)$ defined as $\frac{1}{2} + \frac{1}{4k}$ ($k$ is the degree of symmetry).

Default action for any robot $r$ is to move along its radius to $SEC$ (refer to Figure 3). This default action is chosen if any robot is closer to $c$ than $\mathcal{D}(\cdot)$ or if any non-elected robot is inside $SEC$ or if the robot is not elected. Otherwise the robot goes along its radius to the distance $\mathcal{D}(\cdot)$ from $c$. If and only if all elected robots are at the distance $\mathcal{D}(\cdot)$ from $c$ and all non-elected on $SEC$, every elected robot does a side-step by angle $\mathcal{A}(\cdot)$.

As long as robots are moving only along their radii, the configuration in the PSC model (the projection on $SEC$) is invariant and so are the functions $\mathcal{M}(\cdot)$, $\mathcal{A}(\cdot)$ and $\mathcal{D}(\cdot)$. This corresponds to the observation and computation phase of the PSC model.

The algorithm `Emulate_PSC` ensures that all robots are moving only along their radii until all non-elected robots get on $SEC$ and all elected robots get

[5] We emulate a general algorithm in PSC model and $\mathcal{A}(\cdot)$ is the destination in it.

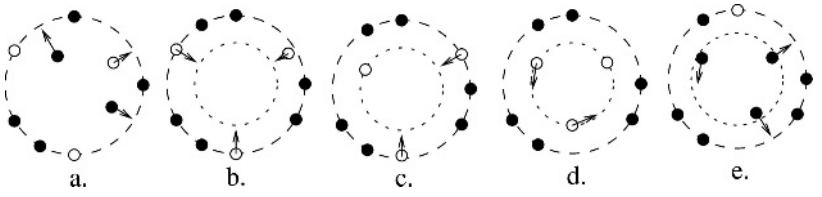


**Fig. 3.** One step – elected robots are white: (a) nonelected robots are inside $SEC$; (b) elected robots go onto the dotted circle; (c) waiting for the synchronized configuration; (d) two elected robots observed the synchronized configurations; (e) $\mathcal{D}(\cdot)$ increased (dotted circle grew), new elected robot waits until all robots get out of the dotted circle and all non-elected on $SEC$

```
EMULATE_PSC(R, r)
 1  if (∃r' ∈ R; |r', c| < D(·))
 2     then return (1, 0)_P
 3  if (∃r' ∈ R \ M(·); r' ∉ SEC)
 4     then return (1, 0)_P
 5  if r ∉ M(·)
 6     then return (1, 0)_P
 7  if (∃r' ∈ M(·); |r', c| ≠ D(·))
 8     then return (D(·), 0)_P
 9  return (D(·), A(r))_P
```

to the distance $\mathcal{D}(\cdot)$ from $c$. We will call this configuration the *synchronized configuration*. At least one elected robot observes the synchronized configuration and starts doing a side-step. Everything from this moment to the moment a new step starts corresponds to the movement phase of the PSC model.

At the begining of each step the following assumptions must hold:

**Assumption 6.** *Every moving robot is moving directly towards $SEC$.*

**Assumption 7.** *Every robot's distance from $c$ is at least $\mathcal{D}(\cdot)$.*

While the Assumption 7 holds, the observation and computation phases of PSC model are emulated (potentially many times again and again). When the Assumption 7 breaks, the movement phase is emulated. When the Assumption 7 becomes valid again, the movement phase has finished and the next step starts. Note that Assumption 6 cannot be checked by observing the configuration and thus cannot be used in the algorithm.

**Correctness of the Emulation.** We have to show that we have synchronized the asynchronous robots into the global steps of the PSC model. Robots must not have remembered any movement across the radii when the next step starts, they can be moving only directly towards $SEC$. In addition we have to show that the next step starts in finite time and assure the progress property of the PSC model.

**Lemma 5.** *When the next step starts, the Assumption 6 holds.*

*Proof.* While at least one elected robot did not start doing a side-step, the actual degree of symmetry $k$ is lower (Assumption 5) than the one in the synchronized configuration and thus the actual $\mathcal{D}(\cdot)$ is more than the one in the synchronized configuration.

While at least one robot is doing a side-step, it is closer to $c$ than in the synchronized configuration because robots are moving along a line. Assumption 5 ensures that $k$ do not increase and thus $\mathcal{D}(\cdot)$ will not decrease.

Robots choose the default action whenever at least one robot is closer to $c$ than $\mathcal{D}(\cdot)$. Every elected robot has to either do a side-step (robots that observed the synchronized configuration) and then take the default action until it gets to the distance $\mathcal{D}(\cdot)$ from $c$ or just take the default action until it gets to the distance $\mathcal{D}(\cdot)$. In both cases every elected robot gets to the distance $\mathcal{D}(\cdot)$ only when it cannot have remembered any side-step movement. This ensures that the Assumption 6 holds when the next step starts.

**Lemma 6.** *The next step starts in a finite time.*

*Proof.* At the start of every step, non-elected robots get in finite time onto $SEC$. Then elected robots move in finite time to the distance $\mathcal{D}(\cdot)$ from $c$. Finally the synchronized configuration is broken and all elected robots get in finite time to the distance at least (actual) $\mathcal{D}(\cdot)$ from $c$ and new step starts.

**Lemma 7.** *At least one robot (if such robot exists) in PSC model passes every step a distance greater than some fixed $\varepsilon$ or get to the destination.*

*Proof.* At least one elected robot will observe the synchronized configuration and start doing a side-step. Assumption 2 ensures that this robot will pass some minimal distance required by PSC model or get to the destination.

### 5.3 Idea of the Solution

We will first use the circle formation algorithm to form a circle. Then we switch to the PSC model and try to transform the circle in a finite sequence of steps to the regular one. We will not be able to achieve it in general and will form only the biangular circle in some cases.

Let us denote one string in the set of the lexicographically first strings of angles LFSA by $S$. Note that the strings in LFSA are the same, they differ only in the starting point and the direction. We have to analyze two cases:

- If all strings in LFSA are oriented in the same direction then only the rotations around $c$ are present. We have the *rotation case* (Figure 4.a) and the string $S$ can be written in form $w^k$ for some $w$.
- If there are strings in LFSA oriented in clockwise and counterclockwise directions then also the mirrorings about axes passing trough $c$ are present. We have the *mirroring case* (Figures 4.b, 4.c) and the string $S$ can be written in form $(ww^R)^{\frac{k}{2}}$ for some $w$, where the first and last angles in $w$ and $w^R$ may

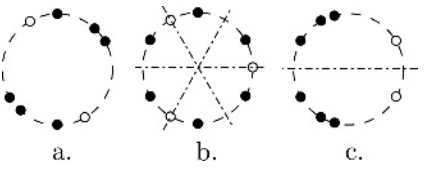


**Fig. 4.** Undistinguishable (white) robots: (a) rotation case, $k = 2$; (b) mirroring case, $k = 6$; (c) mirroring case, $k = 2$

only be half angles of an angle in SA – when the axis of mirroring symmetry passes trough an axis of an angle. Note that in mirroring case the degree of rotation symmetry is $\frac{k}{2}$.

If all elected robots in PSC model are moving synchronously, the degree of symmetry cannot decrease. We will build the regular pattern in each symmetric part of the circle and will not allow the increase of the degree of symmetry until the regular configuration is achieved. The motivation is simple – the solution must consider this synchronous behavior. When the robots break some symmetry, we start building the regular situation from the start with lower degree of symmetry. As the degree of symmetry is a natural number, the process will restart only finite number times. The only exception is the increased degree of symmetry in case the regular circle formation is reached[6].

We have to solve many technical details: We are going to concatenate the circle formation algorithm and the PSC emulation algorithm. We have to perform steps in the PSC model in order to form the regular circle in finite time and assure the invariance of $SEC$ and take care to not increase the degree of symmetry.

### 5.4 Concatenation of Circle Formation and PSC Model

The concatenation of two different algorithms is not possible in general. We have to construct new algorithm, which chooses and calls the right subalgorithm using only the observed configuration.

In our case everything is prepared and the concatenation is done very simply: the circle formation algorithm is used when at least two robots occupy the same radius, the PSC model emulation otherwise.

**Correctness of the Concatenation.** If all robots occupy different radii at the start, only the PSC model emulation algorithm will be used. Otherwise the circle formation algorithm is used first until the last two robots on the same radius are separated. This is done only by robot(s) doing side-steps in the distance not more than $\frac{1}{2}$ from $c$. This is correct start for the PSC model emulation algorithm because only default actions will be performed until the robots finish their side-steps and move along their radii close enough towards $SEC$ to start the first step.

[6] Regular circle will not be formed only when robots are in the moment the circle formation algorithm has finished in the biangular configuration.

### 5.5 Choosing Elected Robots in PSC Model

Assumption 3 gives us the first restriction. In addition we will elect only the smallest possible number of robots.

**Lemma 8.** *The minimal size of a nonempty subset of robots that all robots can agree on from the strings of angles is equal either to* |LFSA| *if no robot is on an axis of a mirroring symmetry, or to* $\frac{|\mathrm{LFSA}|}{2}$ *otherwise.*

*Proof.* The robot $r$ must recognize whether it is elected or not only from its set $\{\mathrm{SA}(r),\ \mathrm{revSA}(r)\}$, because all other robots must agree with it and the other robots do not know its local clockwise orientation.

The robots with equal $\{\mathrm{SA}, \mathrm{revSA}\}$ must be all elected or not, because the robots run the same deterministic algorithm. Thus if a robot $r$ is elected, all other robots with equal SA or revSA must be elected too; $|\{\mathrm{SA}|\ \mathrm{SA} = \mathrm{SA}(r)\}| = |\mathrm{LFSA}|$. Thus the minimal size of a nonempty subset of robots that all robots can agree on is either |LFSA| if $\mathrm{SA}(r) \neq \mathrm{revSA}(r)$, or $\frac{|\mathrm{LFSA}|}{2}$ otherwise.

According to Assumption 3 and Lemma 8 we have to elect one robot for each period $w$ in the rotation case and two symmetric robots (one in the special case) for each rotation period $ww^R$ in the mirroring case. It is therefore sufficient to define the elected robots $\mathcal{M}(\cdot)$ and their moves $\mathcal{A}(\cdot)$ only for the smallest rotation period and this implicitly defines this movement for the whole configuration.

In a biangular configuration (regular being a special case) all robots would be elected and the use of the PSC model emulation would lead to breaking the invariance of SEC. This is why biangular (regular) configurations are handled as a special case – no robot is elected and applying the default action robots move onto $SEC$ and stay there.

### 5.6 Critical Robots

Invariance of $SEC$ during one step is easily achieved when at least one non-identical rotation is among the symmetries (Figures 4.a,b). Any not moving undistinguishable robots on $SEC$ assure its invariance. Otherwise, we cannot send arbitrary robot into $SEC$ without breaking its invariance (Figure 5.b).

We will call a (set of) robot(s) *critical* if its deletion from $SEC$ modifies $SEC$ (see Figures 5.a,c,d, 4.c).

**Lemma 9.** *A robot is critical if and only if the sum of its two adjacent angles is greater than* 180°. *If* $n \geq 4$, *at most two robots can be critical and the critical robots are neighbors.*

*Proof.* Three (two) robots on $SEC$ forming a triangle (line) such that $c$ is inside this triangle (line) are sufficient to assure the invariance of $SEC$. The removal of a robot with the sum of adjacent angles at most 180° thus cannot modify $SEC$.

If there are two not neighboring critical robots, the sum of their adjacent angles would be more than 360°: $2(180° + \varepsilon) > 360°$.

Three neighboring critical robots make similar contradiction (see Figure 5.e): $\alpha + \beta + \gamma + \delta > x + (180° - x) + x + (180° - x) = 360°$.

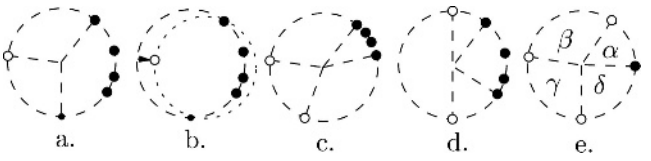


**Fig. 5.** Critical (white) robots cannot leave $SEC$. (a) one critical robot; (b) critical robot breaking the invariance of $SEC$; (c) two critical robots; (d) two critical robots form angle 180°; (e) three robots cannot be critical

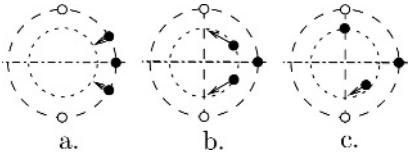


**Fig. 6.** 180° angle is handled as a special case. Critical robots are white. (a) elected robots going to synchronize in distance $\mathcal{D}(\cdot)$; (b) elected robots move to the radii of critical robots; (c) all robots wait until the robot doing side-step finishes

Consider a configuration where two critical robots form with $c$ angle 180° (see Figure 5.c). These two critical robots cannot move. We forbade in the PSC model the robots to stay on the same place and thus no other robot can ever get between the critical robots. We must make an exception.

The nearest robot(s) to the critical ones (with lex. smaller $\{\mathrm{SA}, \mathrm{revSA}\}$, both for equal) will move onto the critical robot(s) in the PSC model[7] (Figure 6). When it (they) get there, circle formation algorithms will be activated and it (they) move between the two critical robots. We achieve it by defining the clockwise direction of the moving robot(s) in the direction to the 180° angle.

**Correctness of the Exception.** The two critical robots assure the invariance of $SEC$. The nearest robot(s) to the critical ones is(are) moving onto it(them), thus it(they) will be the nearest also in the next step. Following Lemma 7 we get that it(they) reach(es) the critical robot(s) in finite time.

The switch to the circle formation algorithm is done in the moment, when the first moving robot $r_1$ reached the destination and the other one $r_2$ (if exist) may be still moving (see figure 6.c). In such a case $r_1$ and all other robots will wait (because $r_2$ is strictly closer to $c$ than any other robot) until $r_2$ either reaches its destination or wakes up and goes directly towards $SEC$. In all cases we have a correct starting situation for the circle formation algorithm. One another robot ($n \geq 5$) always stays on $SEC$ and it ensures that the 180° angle disappears and robots will continue as if there never was any 180° angle.

[7] No robots will meet in reality, only in the projection in $PSC$ model.

### 5.7 Rotation Case

For a given period of angles $w$ we must elect one robot and move it by an angle in order to convert $w$ into the form $\alpha^{|w|}; \alpha = \frac{360^\circ}{kn}$ in finite time. As we do not have to break any symmetries, we can define the first matching robot as the one with the smallest $\{\mathrm{SA}, \mathrm{revSA}\}$.

For our construction we need the configuration with exactly one maximal angle. If this is not the case, we move the first noncritical robot at one side of one of the greatest angles in order to enlarge one maximal angle (Figure 7.a). We will let it pass an angle small enough to not (potentially) increase the degree of symmetry.

Now we know that one angle (mark as $\tau$) in $w$ is the strictly greatest. We will take care to hold the size of all other angles bellow $\tau$. If we choose in the following to move a robot in such a way that $\tau$ could become not the strictly greatest, we move the next one first.

One strictly greatest angle ensures that no rotation symmetry can arise. The only symmetry we will have take care for is the mirroring about the axis running trough the axis of $\tau$ (later only *axis*) (see Figure 7.b).

We will build a sequence of $\alpha$ angles starting at one side of $\tau$. If the sequence of $\alpha$ angles is not complete on either of the axis' sides, we first complete one side. We choose the side of the axis with more robots. If equal number of robots is on both sides, we either move the robot on the axis to one side, either move one robot (closest to the axis or any) onto the axis (see Figure 7.b).

The different number of robots on the sides of the axis ensures that the mirroring symmetry about the axis cannot arise while robots are moving only on one side of the axis. So we can build a sequence of $\alpha$ angles on one of the axis and cram all needless robots between the axis and the last robot in the sequence (see Figures 7.c,7.d). The lengthening of the sequence by one angle is as follows: cram all the abundant robots between the place for the new robot in the sequence and first robot next to this place (b.e. in the middle) or spread them in order to keep all angles bellow $\tau$. Then move the new robot on its place in the sequence.

When the sequence is complete on one side, we continue in building the sequence on the other side (see Figures 7.d-f). We must take care for the mirroring. If the chosen robot's move would increase the degree of symmetry, we first move the next robot a bit (b.e. halfway or less between the original destina-

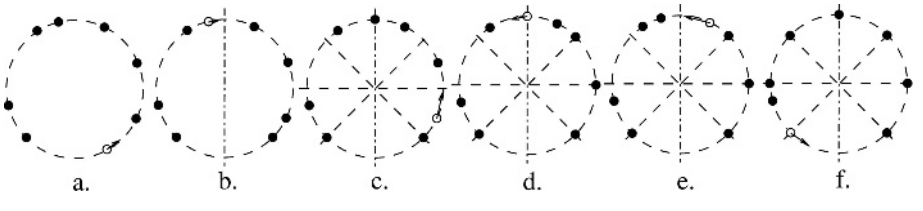


**Fig. 7.** Rotation case. (a) creating one strictly greatest angle; (b) assuring different number of robots on the sides of "axis"; (c)-(f) building a sequence of $\alpha$ angles

tion and next robot's position). At the end the robots stay in the regular circle pattern.

For the correctness observe: the degree of symmetry could not rise until the regular circle pattern is formed, $SEC$ remained invariant, regular circle pattern is formed in finite time.

### 5.8 Mirroring Case

For each string of angles $ww^R$ in a mirroring configuration two symmetric robots (or one on the axis of mirroring) must be elected and the angle they move by must be defined.

The representation $ww^R$ of the period is not sufficient, we also need to know whether there are robots at the beginning and end of $w$ (i.e. whether the robots stay on the intersections of $SEC$ and the axis of mirroring).

When a robot is an the axis of mirroring, we will break the mirroring symmetry and convert the problem to the rotation case in the lower degree of symmetry. The robot on the axis with lex. smaller $\{\mathrm{SA}, \mathrm{revSA}\}$ will move to any side of the axis by an angle one third of the distance to the next robot (see Figure 8a). If this move could rise the degree of symmetry, the robot will move by a smaller angle or to the other side. The side cannot be chosen globally by all robots, but the moving robot can use its local clockwise direction.

So it is sufficient to consider the cases with no robots on the axis of mirroring (and thus $n \geq 6$). Unlike the rotation case, in the mirroring case we cannot start building the sequence of $\alpha$ angles anywhere. We must align the sequence of robots to fit the mirroring symmetry. The nearest robot by the axis will go to the angle $\frac{\alpha}{2}$ from the axis.

To avoid the increasing of the degree of symmetry, we create (in analog to the rotation case) one strictly greatest angle and take care to never increase any other angle to its size. The adjacent half-angles about the axis are considered as one angle.

We will build the sequence of $\alpha$ angles from both sides of $w$. We will lengthen the sequence at that side, where the lengthening will not break the strictly greatest angle and the critical robots need not move. If the sequence can be lengthened on both sides, we continue in the one where the robot dedicated to move is closer to its target. The process of lengthening is the same as in the rotation case.

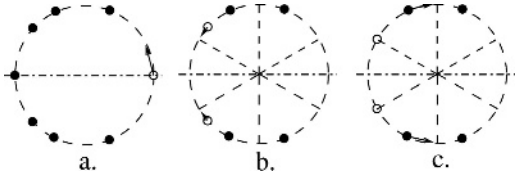


**Fig. 8.** Forming the regular circle in the mirroring case. (a) robot on the axis is breaking the mirroring symmetry; (b),(c) building a sequence of aligned $\alpha$ angles

## 6 Conclusions and Open Issues

We studied in this paper the problem of forming the biangular circle formation. First we presented solution for an easy problem – circle formation. Then we showed that robots can in finite time rearrange the circle to the biangular circle.

The regular circle is not formed only in cases when the biangular configuration is observed in the PSC model. Note that if the robots would have a common sense for clockwise orientation, the electing function $\mathcal{M}(\cdot)$ could be based just on clockwise strings of angles SA. Robots would be able to elect nontrivial subset of them in all cases (except regular circle) and continue in forming the regular circle. This single modification of the algorithm solves the regular circle formation problem in the model with the common sense of clockwise orientation.

Open questions:

- Is it possible to form a regular circle in general case? Try to characterize the configurations transformable into the regular circle.
- Is it possible to form regular circle in the pseudosynchronous model?
- The algorithm supposes infinite precision in the observation. Is it possible to eliminate it? How to define such a model?
- Time complexity of the solution may be defined, analyzed and improved.

*Acknowledgement.* Author would like to thank Rastislav Královič, Dana Pardubská and Michal Forišek for their support.

## References

1. X.Défago, A. Konagaya. Circle formation for oblivious anonymous mobile robots with no common sense of orientation. In Proc. of the 2nd ACM Annual Workshop on Principles of Mobile Computing (POMC'02), pages 97-104, Toulouse, France, October 2002
2. Cieliebak, P. Flocchini, G. Prencipe and N. Santoro. Solving the Robots Gathering Problem. In 30th International Colloquium on Automata, Languages and Programming (ICALP 2003), to appear. Eindhoven, The Netherlands, 30 Giugno – 4 Luglio, 2003.
3. P. Flocchini, G. Prencipe, N. Santoro, and P. Widmayer. Gathering of Asynchronous Mobile Robots with Limited Visibility. In STACS, 2001.
4. 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 ISAAC '99, pages 93-102, 1999
5. I. Chatzigiannakis, M. Marcou, S. Nikoletseas: Distributed circle formation for anonymous oblivious robots. WEA 2004
6. Noa Agmon, David Peleg: Fault-tolerant gathering algorithms for autonomous mobile robots. SODA 2004: 1070-1078
7. Ichiro Suzuki, Masafumi Yamashita: Distributed Anonymous Mobile Robots: Formation of Geometric Patterns. SIAM J. Comput. 28(4): 1347-1363 (1999)