Optimum Gathering of Asynchronous Robots

Subhash Bhagat(✉) and Krishnendu Mukhopadhyaya

ACM Unit, Indian Statistical Institute, Kolkata, India
{sbhagat_r,krishnendu}@isical.ac.in

Abstract. This paper considers the problem of *gathering* a set of asynchronous robots on the two dimensional plane under the additional requirement that the maximum distance traversed by the robots should be minimized. One of the implications of this optimization criteria is the energy efficiency for the robots. The results of this paper are two folds. First, it is proved that multiplicity detection capability is not sufficient to solve the constrained gathering problem for a set of oblivious robots even when the robots are fully synchronous. The problem is then studied for the robots having $O(1)$ bits persistent memory and a distributed algorithm is proposed for the problem in this model for a set of $n \geq 5$ robots. The proposed algorithm uses only two bits of persistent memory.

Keywords: Asynchronous · Gathering · Swarm robots · Robots with lights

1 Introduction

A *swarm* of robots is a distributed system of small, autonomous, inexpensive mobile robots designed to work cooperatively to achieve some goal. They can execute some task which is beyond the capability of a single robot. The system is usually a collection of autonomous (without any centralized control), homogeneous (same capabilities), anonymous (without any identity i.e., indistinguishable) robots. They do not have any explicit communication. The implicit communications are achieved via observing the positions of other robots using their endowed sensors. They do not have any global coordinate system. However, each robot has its own local coordinate system. The directions and orientations of the axes and the unit distances of the local coordinate systems may vary from robot to robot. The robots may be oblivious (they do not remember any information from their past computations).

At any point of time, a robot is either active or inactive (idle). An active robot operates in *Look-Compute-Move* cycles. In the *Look* phase, it takes a snapshot of its surrounding to capture the locations of other robots. In the *Compute* phase, it computes a destination point using information collected in the *Look* phase. Finally, it moves towards the computed destination point in the *Move* phase. An inactive robot does not perform any action i.e., it is in sleep mode. To solve a variety of problems, robots are endowed with some additional capabilities. *Weak multiplicity detection* helps a robot to identify multiple occurrences of robots at

D. Gaur and N.S. Narayanaswamy (Eds.): CALDAM 2017, LNCS 10156, pp. 37–49, 2017.
DOI: 10.1007/978-3-319-53007-9_4

a single point. Whereas, *strong multiplicity detection* enables the robots also to count the total number of robots occupying the same location. The robots may have an agreement on a common orientation (clockwise direction) i.e., *chirality*. They may have some agreement on the directions of their local coordinate axes. In memory model, robots are endowed with externally visible lights, which can assume a constant number of predefined colours, to indicate their states [8,10].

Depending on the timings of the operations and activation schedules of the robots, three types of models are used. The most general model is the *asynchronous (ASYNC or CORDA)* model [14]. The activation of the robots are arbitrary and independent of each other. The time spans of the operations are finite but unpredictable. Thus, robots may compute on some obsolete data. In the *semi-synchronous (SSYNC or ATOM)* [16] model, robots operate in rounds and a subset of robots is activated simultaneously in each round. The operations are instantaneous and hence a robot is not observed while in motion. The unpredictability lies in the activated subset of robots in each round. The most restrictive of these three is the *fully synchronous (FSYNC)* model in which all robots are activated in all rounds. We assume a fair scheduler which activates each robot infinitely often [9].

A variety of problems can be solved by designing proper coordination strategies. Fundamental geometric problems like *gathering*, *circle formation*, *arbitrary pattern formation*, *flocking*, *scattering* etc. have been studied extensively in the literature. The *gathering* problem is defined as follows: a swarm of robots, in the two dimensional plane, should coordinate their motions in such way that in finite time all of them meet at a single point which is not defined in advanced. The *constrained gathering* problem asks robots to achieve gathering by minimizing the maximum distance traversed by any robot.

1.1 Earlier Works

Considering different schedulers and different capabilities of the robots, a variety of solutions have been proposed by researchers for the *gathering* problem. The primary goal of these works has been understanding the minimal set of capabilities of the robots which enables the robots to gather at a point not known in advanced, under different scheduling models.

- *FSYNC Model:* In this model the gathering problem is solvable without any extra assumption [10].
- *SSYNC Model:* Suzuki and Yamashita [17] proved that gathering of $n = 2$ robots is impossible without any agreement on the local coordinate systems even with strong multiplicity detection. Prencipe [15] studied the problem for $n > 2$ robots and proved that there does not exist any deterministic algorithm for the gathering problem in absence of multiplicity detection and any form of agreement on the local coordinate systems [15]. Bramas and Texeuil [3] presented an algorithm to solve the problem in the presence of arbitrary number of crashed robots, when the robots are endowed only with strong multiplicity detection capability. A study of probabilistic gathering was presented by Défago et al. [9].

– *ASYNC Model:* Cieliebak et al. [6] solved the gathering problem for $n > 2$ robots, with weak multiplicity detection capability. Bhagat et al. [2] presented a fault-tolerant distributed algorithm, for $n \geq 2$ robots with agreement in one direction, which solves the problem in the presence of arbitrary number of crashed robots even if robots are opaque i.e., they obstruct the visibility of the other robots. Flocchini et al. [11] showed that gathering is possible in *limited visibility* model (the robots can see up to a limited radius around themselves) if robots have agreements in the directions and orientations of both the axes. The gathering problem for robots, represented as unit discs (*fat robots*), have also been investigated by the researchers [1,4,7]. A restricted version of the gathering problem has been studied in [5] where robots are asked to gather at any one of the predefined fixed points (known as meeting points) on the plane. Their solution also satisfies the constraint that the maximum distance traversed by any robot is minimized. To the best of our knowledge, this is the only work which considers the constrained version of the gathering problem, but for a restricted model. This paper considers the general version of the problem.

The use of externally visible lights was first suggested by Peleg [13]. Das et al. [8] investigated the characterizations of the model, in which robots are endowed with externally visible lights. In memory model, the gathering problem for $n = 2$ robots (also known as *rendezvous*), was studied under different restrictions. The studies of [18] and [8] proposed solutions to the *rendezvous* problem, using the lights for both internal memory and communication purpose. Flocchini et al. [12] investigated the possibilities of solving the *rendezvous* problem in two models: (i) the lights are used only to remember internal states and (ii) light are used for communication purposes.

1.2 Our Contribution

In this paper, we study the constrained gathering problem for a set of autonomous robots. The contribution of this paper is in two parts. While the gathering problem is solvable for $n > 2$ asynchronous robots with weak multiplicity detection only, it is shown that even in the $FSYNC$ model, multiplicity detection capability is not sufficient to solve the constrained gathering problem for oblivious robots. A distributed algorithm is then presented to solve the problem in finite time for a set of $n \geq 5$ asynchronous, oblivious robots under the assumptions that robots are endowed with only two bits of memory. We do not make any extra assumption like agreements in coordinate systems, unit distance and chirality, rigidity of movements. In spite of these weak assumptions, we have showed that the constrained gathering problem is solvable for asynchronous robots using only four colours. Our solution also provides collision free movements for the robots. To the best of our knowledge, this paper is the first attempt to study the constrained gathering problem, in general, for asynchronous robots. One may view this constrained version of gathering problem as a solution to energy efficiency.

2 General Model and Definitions

The robots are autonomous, homogeneous, anonymous, asynchronous in nature. They are considered as points in the two dimensional plane and they can move freely on the plane. We consider the $ASYNC$ model ($CORDA$). Each robot has its own local coordinate system (Cartesian coordinate system) having origin at its current position. The directions and orientations of the axes and unit distances of the local coordinate systems may differ. They do not have any common chirality. All the measurements are done with respect to the local coordinate systems of the robots. We assume that initially all the robots occupy distinct points. The visibility range of the robots is unlimited. The movements of the robots are non-rigid i.e., a robot may stop before reaching its destination point and start a fresh computational cycle. However, there exists a constant $\delta > 0$ such that it moves at least a distance $minimum\{\delta, d\}$ towards its destination point where d is the distance of its destination point from its current position. It assures finite time reachability of the robots to their respective destinations. The value of δ is not known to the robots. Since the timing of operations by the robots are unpredictable, a robot may be observed by other robots in the system while it is in motion and the computation of robot may be done on some obsolete data.

- **Configuration of the Robots:** Let $\mathcal{R} = \{r_1, r_2, \ldots, r_n\}$ denote the set of n robots. A robot configuration is denoted by the multi set $\mathcal{R}(t) = \{r_1(t), \ldots, r_n(t)\}$, where $r_i(t)$ is the position of the robot r_i at time t. Let $\widetilde{\mathcal{R}}$ denote the set of all such robot configurations. We assume that the initial configuration $\mathcal{R}(t_0)$ does not contain any multiplicity point (a point having multiple robots on it).
- **Smallest Enclosing Circle:** Let $SEC(\mathcal{R}(t))$ denote the smallest enclosing circle of the points in $\mathcal{R}(t)$ and $\mathcal{O}_t$ denote its centre. Let $C_{out}(t)$ denote the set of robot positions on the circumference of $SEC(\mathcal{R}(t))$ and $C_{int}(t)$ the set of robot positions lying within $SEC(\mathcal{R}(t))$. When there is no ambiguity, we use $SEC(t)$ instead of $SEC(\mathcal{R}(t))$.
- Let $\overline{ab}$ denote the closed line segment joining two points a and b (including the end points a and b) and (a, b) the open line segment joining the points a and b (excluding the two end points a and b). By $|a, b|$, we denote the distance between the points a and b. For two sets A and B, by $A \backslash B$ we denote the set difference of A and B. The angle between two given line segments is considered as the angle which is less than or equal to π.

We use the following terms, as defined in [3], to describe our algorithm:

- **View of a Robot:** For a robot $r_i \in \mathcal{R}$, the view $\mathcal{V}(r_i(t))$ of r_i is defined as the set of polar coordinates of the points in $\mathcal{R}(t)$ where the polar coordinate system of r_i is defined as follows: (i) the centre of the coordinate system is $r_i(t)$ and (ii) the point $(1, 0)$ is $\mathcal{O}_t$ if $r_i(t) \neq \mathcal{O}_t$, otherwise it is any point $r_k(t) \neq r_i(t) \in \mathcal{R}(t)$ that maximizes $\mathcal{V}(r_k(t))$. The orientation of the polar coordinate system should maximize $\mathcal{V}(r_i(t))$. The view of each robot is defined uniquely. While comparing the views of two robots, lexicographic sorting is used.

- **Rotational Symmetry:** An equivalence relation $\sim$ is defined on $\mathcal{R}(t)$ as follows: $\forall r_i(t), r_j(t) \in \mathcal{R}(t)$, $r_i(t) \sim r_j(t)$ iff $\mathcal{V}(r_i(t)) = \mathcal{V}(r_j(t))$ with same orientation. Let $sym(\mathcal{R}(t))$ denote the cardinality of the largest equivalence class defined by $\sim$. The set $\mathcal{R}(t)$ is said to have rotational symmetry if $sym(\mathcal{R}(t)) > 1$.
- **Successor:** Given a robot configuration $\mathcal{R}(t)$ and a fixed point $c \in \mathbb{R}^2$, the clockwise successor of a point $r_i(t) \in \mathcal{R}(t)$ around c, denoted by $S(r_i(t), c)$, is the point $r_j(t) \in \mathcal{R}(t)$ defined as follows: if $(c, r_i(t))$ contains at least one point of $\mathcal{R}(t)$, then $r_j(t)$ is the point in $\mathcal{R}(t) \cap (c, r_i(t))$ which minimizes $|r_i(t), r_j(t)|$. Otherwise, $r_j(t)$ is the point in clockwise direction such that $\angle(r_i(t), c, r_j(t))$ contains no other point of $\mathcal{R}(t)$ and $|c, r_j(t)|$ is maximized. The counter-clockwise successor of r_i can be defined analogously. The k^{th} *clockwise successor* of $r_i(t)$ around c, denoted by $S^k(r_i(t), c)$, is defined by the recursive relation: *for* $k > 1$, $S^k(r_i(t), c) = S(S^{k-1}(r_i(t), c))$, $S^1(r_i(t), c) = S(r_i(t), c)$ and $S^0(r_i(t), c) = r_i(t)$.
- **String of Angles:** Let $SA(r_i(t), c)$ denote the string of angles $\alpha_1(t), \alpha_2(t), \ldots, \alpha_m(t)$ where $m = n - mult(c)$, $mult(c)$ is the number of robots at the point c and $\alpha_i(t) = \angle(S^{i-1}(r_i(t)), c, S^i(r_i(t)))$. The length of $SA(r_i(t), c)$ is $|SA(r_i(t), c)| = m$. The string $SA(r_i(t), c)$ is *k-periodic* if there exists a constant $1 \leq k \leq m$ such that $SA(r_i(t), c) = X^k$, where X is a sub-string of $SA(r_i(t), c)$. The *periodicity* of $SA(r_i(t), c)$, denoted by $per(SA(r_i(t), c))$, is the largest value of k for which $SA(r_i(t), c)$ is *k-periodic*.
- **Regularity:** A robot configuration $\mathcal{R}(t)$ is said to be *regular* if $\exists$ a point $c \in \mathbb{R}^2$ and an integer m such that $per(SA(r_i(t), c)) = m > 1$ and the *regularity* of $\mathcal{R}(t)$ is denoted by $reg(\mathcal{R}(t)) = m$. The point c is called the centre of regularity.
- **Quasi Regularity:** A robot configuration $\mathcal{R}(t)$ is said to be *quasi regular* or *Q-regular* iff $\exists$ a configuration $\mathcal{B}(t)$ and a point $c \in \mathbb{R}^2$ such that $reg(\mathcal{B}(t)) > 1$, c is the centre of regularity of $\mathcal{B}(t)$ and $p \in \mathcal{R}(t) \backslash \mathcal{B}(t)$, $p = c$. In other words, $\mathcal{B}(t)$ can be obtained from $\mathcal{R}(t)$ by moving the robot positions located at c, if any, along particular half lines starting at c (including c). If $\mathcal{R}(t)$ is Q-regular, then the point c is called as the centre of Q-regularity. By c_q, we denote the centre of Q-regularity. The quasi-regularity of $\mathcal{R}(t)$ is denoted $qreg(\mathcal{R}(t)) = reg(\mathcal{B}(t))$. If $\mathcal{R}(t)$ is not quasi-regular, then $qreg(\mathcal{R}(t)) = 1$.

Fact 1. *Let $\mathcal{R}(t_0)$ be an initial robot configuration. Then the centre $\mathcal{O}_{t_0}$ of $SEC(t_0)$ is the unique point which minimizes the maximum distance from any point in $\mathcal{R}(t_0)$ to it.*

For a set of points $\mathcal{A}$, the *Weber point* of $\mathcal{A}$ is a point c which minimizes the sum of distances of all the points in $\mathcal{A}$ to it. It is known that the Weber point is not computable, in general, for a set of more than four points.

Fact 2. *Weber point of a non-linear configuration is unique and this point remains invariant under the straight movements of the robots towards it.*

Fact 3. *If a configuration has multiple lines of symmetry, then it is Q-regular. However, the converse is not true.*

Fact 4. *The centre of Q-regularity of a non-linear configuration coincides with its unique Weber point and it is computable in finite time.*

Fact 5. *For a set A of $n \geq 3$ points, there exists a subset $B \subseteq A$ such that $|B| \leq 3$ and the smallest enclosing circles of A and B are same.*

We state the following theorem without proof:

Theorem 1. *The constrained gathering problem for a set of oblivious robots is deterministically unsolvable even with strong multiplicity detection capability under the FSYNC model.*

3 Gathering Algorithm with Persistent Memory

By Fact 1, for an initial configuration $\mathcal{R}(t_0)$, the centre $\mathcal{O}_{t_0}$ of the circle $SEC(\mathcal{R}(t_0))$ is the only candidate for the gathering point which satisfies the optimization criteria. But this point does not remain invariant under the movements of the robots. Thus, strategies are designed in such a way that even when configuration is changed, the robots can identify the point $\mathcal{O}_{t_0}$. We assume that each robot has two bits of persistent memory. This is implemented by endowing the robots with externally visible lights. These lights can assume a finite number of colours, each colour indicates a different state. The colours do not change automatically and they are persistent. The lights are used in two different ways: one way is to remember the robot's own state and the other way is to broadcast its current state i.e., for both internal memory and communication purpose [12]. A robot can identify the colours of the lights of all the robots in the system even when multiple robots occupy same position. It may be noted that strong multiplicity detecting follows directly from it; the light model provides some additional powers to the robots also. Except for the colour of its light, robots are oblivious i.e., they do not remember any other information of its previous computational cycles. Our algorithm assumes total four colours i.e., two bits memory.

3.1 States of the Robots

The robots use colours for their external lights to indicate their current states. This set of colours is denoted by $\mathcal{X}$. Let $\mathcal{R}(t_0)$ be an initial configuration. Following are the list of states and their corresponding colours: *yellow* indicates that it is an inactive robot or it has not changed its colour yet, *red* indicates that it has found $\mathcal{R}(t_0)$ as a Q-regular configuration in which the centre of $SEC(\mathcal{R}(t_0))$ coincides with centre of Q-regularity i.e., $\mathcal{O}_{t_0} = c_q$, *green* indicates that it has found either $\mathcal{R}(t_0)$ as not Q-regular or as Q-regular with $\mathcal{O}_{t_0} \neq c_q$ and initially the robot was not at $\mathcal{O}_{t_0}$ and *blue* indicates that it has found either

$\mathcal{R}(t_0)$ as not Q-regular or as Q-regular with $\mathcal{O}_{t_0} \neq c_q$ and it is at $\mathcal{O}_{t_0}$. Thus, $\mathcal{X} = \{yellow, red, green, blue\}$. Let $s_i(t)$ denote the colour of the light for the robot r_i at time t.

3.2 Configurations

Let $\mathcal{R}_c(t)$ denote the set of all tuples $(r_i(t), s_i(t))$ for all $r_i(t) \in \mathcal{R}(t)$ and $s_i(t) \in \mathcal{X}$. The set of all such $\mathcal{R}_c(t)$ is denoted by $\widetilde{\mathcal{R}_c}$ where $\mathcal{R}(t) \in \widetilde{\mathcal{R}}$. We partition $\widetilde{\mathcal{R}_c}$ into the following classes:

- **Central ($\mathcal{CL}$):** A robot configuration $\mathcal{R}_c(t)$ belongs to this class if it satisfies exactly one of the following: (i) $\mathcal{R}_c(t)$ contains a multiplicity point p_m such that at least one robot at p_m has *red* light or (ii) $\mathcal{R}_c(t)$ contains a point such that at least one robot at this point has *blue* light. A configuration in this class is called a central configuration and the multiplicity point or the point with blue robot is called the central point.
- **Q*-regular ($\mathcal{QR}_0$):** A robot configuration $\mathcal{R}_c(t)$, containing no multiplicity point, is in this class if (i) $\mathcal{R}(t)$ is Q-regular with $\mathcal{O}_t = c_q$ and all the robots have *yellow* light or (ii) contains at least one tuple $(r_i(t), s_i(t))$ such that the value of $s_i(t)$ is *red*.
- **Non Q*-regular ($\mathcal{NQ}$):** This class contains all the configurations $\mathcal{R}(t) \in \widetilde{\mathcal{R}_c} \backslash (\mathcal{CL} \cup \mathcal{QR}_0)$. The configurations which are asymmetric or have exactly one line of symmetry belong to this class.

3.3 Algorithm *MoveToDestination*()

Let r_i be a robot which has a destination point p_x. The robot r_i follows following steps to reach the point p_x:

- If $(r_i(t), p_x) \cap \mathcal{R}(t) = \emptyset$ i.e., there is no other robot position in between $r_i(t)$ and p_x on the line segment $\overline{r_i(t)p_x}$, the robot r_i moves directly to p_x along $\overline{r_i(t)p_x}$.
- If $(r_i(t), p_x) \cap \mathcal{R}(t) \neq \emptyset$ i.e., there is at least one robot position in between $r_i(t)$ and p_x on the line segment $\overline{r_i(t)p_x}$, the robot r_i waits until it finds a free corridor straight to p_x.

3.4 Algorithm *GatheringLight*()

We assume that (i) the initial robot configuration $\mathcal{R}(t_0)$ does not contain any multiplicity point (ii) all the robots initially have *yellow* lights and (iii) $n \geq 5$. An initial configuration $\mathcal{R}_c(t_0)$ belongs to either $\mathcal{QR}_0$ or $\mathcal{NQ}$. The basic idea is to convert the initial configuration, within finite time, into a central one i.e., one in $\mathcal{CL}$ in which $\mathcal{O}_{t_0}$ is the central point. During the conversion phase, the movements of the robots are coordinated in such way that the initial $SEC(t_0)$ does not change until a central configuration is created. Once a central configuration is created, the point $\mathcal{O}_{t_0}$ remains recognizable by the robots even if the

initial $SEC(t_0)$ changes. The movements of the robots are designed to satisfy the constraint of the problem. Let r_i be an arbitrary active robot in $\mathcal{R}$, at time t. Robot r_i takes one of the following actions, depending on the configuration and the position of the robot:

- **Case-1 $\mathcal{R}(t) \in \mathcal{QR}_0$:** In this case, all the active robots have c_q as their destination point. Robot r_i finds that either all robots have *yellow* lights or at least one robot has *red* light. If $r_i(t) = c_q$, the robot r_i does not move. Otherwise, it does one of the following: (i) if there is a *yellow* robot at c_q, the robot r_i waits (ii) otherwise, it sets c_q as its destination point. In all these sub-cases, if the light of r_i is *yellow*, it also changes the colour to *red*.
- **Case-2 $\mathcal{R}(t) \in \mathcal{NQ}$:** The robot r_i finds either all robots with *yellow* light or at least one robot with *green* light and it acts according to the following:
  - **Case-2.1 $C_{int}(t) \neq \emptyset$:** If $r_i(t) \in C_{out}(t)$, the robot r_i does nothing. Otherwise, it does one of the following: (i) if $r_i(t) = \mathcal{O}_t$, the robot r_i does not move and it turns its light *blue*. (ii) if $r_i(t) \neq \mathcal{O}_t$, the robots r_i sets $\mathcal{O}_t$ as its destination and it changes its colour to *green*.
  - **Case-2.2 $C_{int}(t) = \emptyset$:** In this case all the robots lie on the circumference of $SEC(t)$.
    - **Case-2.2.1 $\mathcal{R}(t)$ is not Q-regular:** Fact 3 implies that the robot positions in $\mathcal{R}(t)$ have at most one line of symmetry.
      - **Case-2.2.1.1 $\mathcal{R}(t)$ does not have any line of symmetry:** The robot positions in $\mathcal{R}(t)$ are orderable in this case [4]. Consider an ordering $\mathcal{G}$ (the ordering algorithm is same for all robots) of the points in $\mathcal{R}(t)$. Select the robot position $r_u(t) \in \mathcal{R}(t)$ such that the smallest enclosing circle of $\mathcal{R}(t) \backslash \{r_u(t)\}$ is same as $SEC(\mathcal{R}(t))$ and $r_u(t)$ has the highest order in $\mathcal{G}$ among all the points satisfying this property. Since $n \geq 5$, such a point exists by Fact 5. If $r_i(t) = r_u(t)$, then it moves towards $\mathcal{O}_t$ and turns its light *green*. Otherwise, it does nothing.
      - **Case-2.2.1.2 $\mathcal{R}(t)$ has exactly one line of symmetry $\mathcal{L}$**: There are two possibilities: (i) the line $\mathcal{L}$ passes through at least one robot position in $\mathcal{R}(t)$ or (ii) there are four robot positions, say $H_1 = \{r_{v_1}(t), r_{v_2}(t), r_{v_3}(t), r_{v_4}(t)\}$, belonging to $C_{out}(t)$ which are closest to $\mathcal{L}$. If $r_i(t) \in H_1$ or $\mathcal{L}$ does not pass through $r_i(t)$, the robot r_i does nothing. Otherwise, r_i has $\mathcal{O}_t$ as its destination point and it changes its colour to *green*.
    - **Case-2.2.2 $\mathcal{R}(t)$ Q-regular with $\mathcal{O}_t \neq c_q$:** There are at most two distinct robot positions in C_{out} which are farthest from c_q. Let $\mathcal{U}$ denote the set of these points. $\mathcal{L}_z$ is the ray defined as follows (i) if $|\mathcal{U}| = 1$ and $r_l(t) \in \mathcal{U}$, then $\mathcal{L}_z$ is the ray from $r_l(t)$, which passes through $\mathcal{O}_t$ (ii) otherwise, it is the ray from the middle point of the two robot positions in $\mathcal{U}$, which passes through $\mathcal{O}_t$. Let $\mathcal{L}_z$ intersect the circumference of $SEC(t)$ at p_z. $\mathcal{W}$ is the set defined as follows: (i) if p_z contains a robot position, $\mathcal{W}$ is the singleton set containing this

point (ii) otherwise, $\mathcal{W} = \{r_j(t), r_k(t)\}$ where $r_j(t), r_k(t) \in C_{out}(t)$ and they lie on two different sides of $\mathcal{L}_z$. If $r_i(t) \neq \mathcal{U} \cup \mathcal{W}$, it sets $\mathcal{O}_t$ as its destination point and turns its light *green*. Otherwise, the robot r_i does nothing. Note that $|\mathcal{U} \cup \mathcal{W}|$ is at most 4 and the robot positions in $\mathcal{U} \cup \mathcal{W}$ keep $SEC(t)$ intact. Since $n \geq 5$, at least one robot moves inside $SEC(t)$.

– **Case-3 $\mathcal{R}(t) \in \mathcal{CL}$:** An initial configuration $\mathcal{R}_c(t_0)$ does not belong to this class. A configuration in this class is generated by the movements of the robots as described in case-1 and case-2. The destination point of each robot is $\mathcal{O}_{t_0}$. First, consider the case when $\mathcal{R}_c(t)$ contains a multiplicity point at $\mathcal{O}_{t_0}$, with all robots at this point having *red* colour. If $r_i(t) = \mathcal{O}_{t_0}$, the robot r_i does not move. Otherwise, it sets $\mathcal{O}_{t_0}$ as its destination point. In both the cases, if the colour of r_i is *yellow*, it changes its colour to *red*. Since robots can distinguish colours of all the robots occupying the same position, they can easily identify the point $\mathcal{O}_{t_0}$. Now, let $\mathcal{R}_c(t)$ contain a *blue* robot at the point $\mathcal{O}_{t_0}$. Same strategies are followed. In this case a robot does not change its colour.

In all of the above cases, a robot moves to its destination point according to algorithm *MoveToDestination*() described in Sect. 3.3. Following list shows the transactions between different states of the robots: $\{yellow\} \xrightarrow{\mathcal{R}(t) \in \mathcal{QR}_0} \{red\}$, $\{yellow\} \xrightarrow{\mathcal{R}(t) \in \mathcal{NQ} \wedge C_{int}(t) \neq \emptyset \wedge \mathcal{O}_t = r_i(t)} \{blue\}$, $\{yellow\} \xrightarrow{\mathcal{R}(t) \in \mathcal{NQ} \wedge \mathcal{O}_t \neq r_i(t)} green$, $\{green\} \xrightarrow{\mathcal{R}(t) \in \mathcal{NQ} \wedge C_{int}(t) \neq \emptyset \wedge \mathcal{O}_t = r_i(t)} \{blue\}$.

3.5 Correctness of *GatheringLight*()

In this section, it is proved that *GatheringLight*() solves the constrained gathering problem.

Lemma 1. *Algorithm MoveToDestination() guarantees collision free movements for the robots during the whole execution of algorithm GatheringLight().*

Proof. During the whole execution of algorithm *GatheringLight*(), the destination point for each robot in the system is $\mathcal{O}_t$. Let r_i be robot which wants to move to the point $\mathcal{O}_t$. The robot r_i starts moving towards $\mathcal{O}_t$ only when it finds a free corridor straight to this point. Otherwise, it waits, until all the robots in $(r_i(t), \mathcal{O}_t)$ reach their respective destinations. Thus, algorithm *MoveToDestination*() guarantees a collision free movement for the robot r_i. □

Lemma 2. *During the whole execution of algorithm GatheringLight(), $\nexists$ a time t such that $s_i(t) = red$ and $s_j(t) = green \vee blue$ for any two robots $r_i, r_j \in \mathcal{R}$.*

Proof. Let $\mathcal{R}(t_0)$ be an initial robot configuration. Initially all the robots have *yellow* lights.

- **Case-1 $\mathcal{R}_c(t_0) \in \mathcal{QR}_0$:** Since $\mathcal{O}_{t_0} = c_q$, when a robot wakes up, it either finds that all robots are *yellow* or at least one robot has *red* light. In both the cases, it turns its light *red* and it never changes its colour again during the whole execution of the algorithm. Thus through out the whole execution of the algorithm, each robot has any one of the colours from the set $\mathcal{F}_1 = \{yellow, red\}$ for its light.
- **Case-2 $\mathcal{R}_c(t_0) \in \mathcal{NQ}$:** The robots which wake up first, find that $\mathcal{R}(t_0) \in \mathcal{NQ}$ and all the robots have *yellow* lights. First, they turn their lights *green* or *blue*, depending upon their positions and then execute *move* phase. The robots which wake up after this, find that either $\mathcal{R}_c(t)$ has at least one robot with *green* light or it contains unique point occupied by at least one *blue* robot. They change their colours to *green* or to *blue*. Hence, in this case, the set of colours of consumed by the robots is $\mathcal{F}_2 = \{yellow, green, blue\}$.
- **Case-3 $\mathcal{R}_c(t) \in \mathcal{CL}$:** This case is applicable for a robot configuration $\mathcal{R}_c(t)$ where $t > t_o$. From case-1 and case-2, it is clear that $\mathcal{R}_c(t)$ does not contain two robots; one with *red* colour and other with *blue* colour. When a robot finds a multiplicity point with *red* robots, it changes its colour to *red*. Otherwise, when it finds a robot with *blue* colour, it does not change its colour. This implies at time $t' > t$, the system does not have two robots; one with *red* light and other with *blue* light. Hence the lemma holds. □

Lemma 3. *Algorithm GatheringLight() converts any initial configuration $\mathcal{R}(t_0)$ with more than 4 distinct robot positions, to a central configuration in finite time.*

Proof. Let $\mathcal{R}_c(t_0) \notin \mathcal{CL}$ be an initial robot configuration with more than 4 distinct robot positions. The algorithm *GatheringLight*() maintains one of the two invariants (i) if the initial configuration is Q-regular with $\mathcal{O}_{t_0} = c_q$, it remains Q-regular until a multiplicity point with *red* robots is created (ii) otherwise, the smallest enclosing circle $SEC(t_0)$ of $\mathcal{R}(t_0)$ remains the same until at least one robot reaches $\mathcal{O}_{t_0}$ and changes its colour to *blue*. Thus each active robot has exactly one desired destination point $\mathcal{O}_{t_0}$.

- **Case-1 $\mathcal{R}(t_0) \in \mathcal{QR}_0$:** Here, $\mathcal{O}_{t_0} = c_q$ and the robots move towards c_q. By Fact 2, the point c_q, remains invariant under the straight movements of the robots towards it. The robots which are moving towards c_q, have *red* colour lights to indicate the state of the initial configuration. By Lemma 2, within finite time at least two *red* robots occupy the point $\mathcal{O}_{t_0} = c_q$. This converts c_q as a multiplicity point with all *red* robots. The uniqueness of the multiplicity point follows from Lemma 1. Thus, within finite time, we would have a central configuration.
- **Case-2 $\mathcal{R}(t_0) \in \mathcal{NQ}$:** The robots which discover this case first, turn their lights *green* or *blue* so that all other robots, whenever they wake up, can have this information about the initial configuration. The robots have any one of the following scenarios:
  - **Case-2.1 $C_{int}(t_0) \neq \emptyset$:** In this case, the robots in $C_{int}(t_0)$ move towards $\mathcal{O}_{t_0}$. Since the robots in $C_{out}(t_0)$ do not move, the circle $SEC(t_0)$ and

hence $\mathcal{O}_{t_0}$, remain invariant. Within finite time at least one robot reaches $\mathcal{O}_{t_0}$ and turns its light *blue*, which converts the configuration into a central one. Since a robot at $\mathcal{O}_{t_0}$ decides to turn its colour to *blue* when it finds no robot with *blue* light in the system, the point with *blue* robots is unique.

- **Case-2.2 $C_{int}(t_0) = \emptyset$:** The objective, in this case, is to move at least one robot inside $SEC(t_0)$ so that $|C_{int}(t)|$ becomes at least 1. The selection of this robot depends on the symmetry of $\mathcal{R}(t_0)$.
  - **Case-2.2.1 $\mathcal{R}(t_0)$ is not Q-regular:** There are two possibilities:
    - **Case-2.2.1.1 $\mathcal{R}(t_0)$ does not have any line of symmetry:** The robot positions in $\mathcal{R}(t_0)$ are orderable. Exactly one robot can be selected deterministically to move inside $SEC(t_0)$ such that $SEC(t_0)$ remains invariant.
    - **Case-2.2.1.2 $\mathcal{R}(t_0)$ has exactly one line of symmetry $\mathcal{L}$:** Since $n \geq 5$ and at most four robots retain their positions on the circumference of $SEC(t_0)$ to keep it intact, there is at least one robot which is eligible to move inside $SEC(t_0)$. Whenever one such robot moves inside $SEC(t_0)$, we are done.
  - **Case-2.2.2 $\mathcal{R}(t_0)$ is Q-regular with $\mathcal{Q}_{t_0} \neq c_q$:** Same arguments as in the case-2.2.1.2 above, works for this case also.

  Within finite time, $|C_{int}(t)|$ will be at least 1 and case-2.1 shall be applicable. This implies that the initial configuration will be converted into a central one within finite time.

Hence, the lemma is true. □

Lemma 4. *Algorithm GatheringLight() solves the constrained gathering problem in the ASYNC model for $n \geq 5$ robots in finite time.*

Proof. By Lemma 3, an initial configuration $\mathcal{R}_c(t_0)$ can be converted into a central configuration in finite time if the number of distinct robot positions in the configuration is more than 4. Now, let $\mathcal{R}(t) \in \mathcal{CL}$, $\forall\, t > t_0$. By Lemma 2, exactly one of the following is the case: (i) $\mathcal{R}(t)$ contains a unique multiplicity point with *red* robots or (ii) $\mathcal{R}(t)$ contains a unique point with *blue* robots. In both the cases, the special point is created at $\mathcal{O}_{t_0}$. Thus, a robot not at $\mathcal{O}_t$ can easily identify the location of $\mathcal{O}_{t_0}$ even when the circle $SEC(t_0)$ has changed. All the robots move towards $\mathcal{O}_{t_0}$. They follow algorithm $MoveToDestination()$ to reach their destination point. Algorithm $MoveToDestination()$ guarantees that (i) the movements of the robots are collision free, by Lemma 1 and (ii) the movements of the robots satisfy the constraint of the problem (since robots move along the straight lines to the destination points). Thus, algorithm *GatheringLight*() achieves the required goal. It is easy to see that the constrained gathering problem is not solvable, in general, for $n \leq 4$ robots with lights. □

Theorem 2. *The constrained gathering problem is solvable in the ASYNC model for $n \geq 5$ robots when robots are endowed with externally visible lights with 4 different colours.*

4 Conclusion

This paper presents a distributed algorithm to solve the constrained version of gathering problem for asynchronous robots, when robots are endowed with externally visible lights using only 4 colours ($O(1)$ bit of memory) for $n \geq 5$ robots. It is also proved that the problem is not solvable solely with multiplicity detection under non-rigid motion even for fully synchronous robots. One of the future directions is to study the problem in the presence of faulty robots.

References

1. Agathangelou, C., Georgiou, C., Mavronicolas, M.: A distributed algorithm for gathering many fat mobile robots in the plane. In: Proceedings of ACM Symposium on Principles of Distributed Computing (PODC), pp. 250–259 (2013)
2. Bhagat, S., Chaudhuri, S.G., Mukhopadhyaya, K.: Fault-tolerant gathering of asynchronous oblivious mobile robots under one-axis agreement. J. Discret. Algorithms **36**, 50–62 (2016)
3. Bramas, Q., Tixeuil, S.: Wait-free gathering without chirality. In: Scheideler, C. (ed.) Structural Information and Communication Complexity. LNCS, vol. 9439, pp. 313–327. Springer, Heidelberg (2015). doi:10.1007/978-3-319-25258-2_22
4. Chaudhuri, S.G., Mukhopadhyaya, K.: Leader election and gathering for asynchronous fat robots without common chirality. J. Discret. Algorithms **33**, 171–192 (2015)
5. Cicerone, S., Di Stefano, G., Navarra, A.: Minmax-distance gathering on given meeting points. In: Paschos, V.T., Widmayer, P. (eds.) CIAC 2015. LNCS, vol. 9079, pp. 127–139. Springer, Heidelberg (2015). doi:10.1007/978-3-319-18173-8_9
6. Cieliebak, M., Flocchini, P., Prencipe, G., Santoro, N.: Distributed computing by mobile robots: gathering. SIAM J. Comput. **41**(4), 829–879 (2012)
7. Czyzowicz, J., Gasieniec, L., Pelc, A.: Gathering few fat mobile robots in the plane. Theor. Comput. Sci. **410**(6), 481–499 (2009)
8. Das, S., Flocchini, P., Prencipe, G., Santoro, N., Yamashita, M.: The power of lights: synchronizing asynchronous robots using visible bits. In: Proceedings of IEEE 32nd International Conference on Distributed Computing Systems (ICDCS), pp. 506–515 (2012)
9. Défago, X., Gradinariu, M., Messika, S., Raipin-Parvédy, P.: Fault-tolerant and self-stabilizing mobile robots gathering. In: Dolev, S. (ed.) DISC 2006. LNCS, vol. 4167, pp. 46–60. Springer, Heidelberg (2006). doi:10.1007/11864219_4
10. Flocchini, P., Prencipe, G., Santoro, N.: Distributed computing by oblivious mobile robots. In: Synthesis Lectures on Distributed Computing Theory. Morgan & Claypool Publishers (2012)
11. Flocchini, P., Prencipe, G., Santoro, N., Widmayer, P.: Gathering of asynchronous robots with limited visibility. Theor. Comput. Sci. **337**(1–3), 147–168 (2005)
12. Flocchini, P., Santoro, N., Viglietta, G., Yamashita, M.: Rendezvous of two robots with constant memory. In: Moscibroda, T., Rescigno, A.A. (eds.) SIROCCO 2013. LNCS, vol. 8179, pp. 189–200. Springer, Heidelberg (2013). doi:10.1007/978-3-319-03578-9_16
13. Peleg, D.: Distributed coordination algorithms for mobile robot swarms: new directions and challenges. In: Pal, A., Kshemkalyani, A.D., Kumar, R., Gupta, A. (eds.) IWDC 2005. LNCS, vol. 3741, pp. 1–12. Springer, Heidelberg (2005). doi:10.1007/11603771_1

14. Prencipe, G.: *Instantaneous actions* vs. *full asynchronicity*: controlling and coordinating a Sset of autonomous mobile robots. In: Restivo, A., Della Rocca, S.R., Roversi, L. (eds.) ICTCS 2001. LNCS, vol. 2202, pp. 154–171. Springer, Heidelberg (2001). doi:10.1007/3-540-45446-2_10
15. Prencipe, G.: Impossibility of gathering by a set of autonomous mobile robots. Theor. Comput. Sci. **384**(2–3), 222–231 (2007)
16. Suzuki, I., Yamashita, M.: Formation and agreement problems for anonymous mobile robots. In: Proceedings of 31st Annual Conference on Communication, Control and Computing, pp. 93–102 (1993)
17. Suzuki, I., Yamashita, M.: Distributed anonymous mobile robots: formation of geometric patterns. SIAM J. Comput. **28**, 1347–1363 (1999)
18. Viglietta, G.: Rendezvous of two robots with visible bits. In: Flocchini, P., Gao, J., Kranakis, E., Meyer auf der Heide, F. (eds.) ALGOSENSORS 2013. LNCS, vol. 8243, pp. 291–306. Springer, Heidelberg (2014). doi:10.1007/978-3-642-45346-5_21