# Bounds on Mutual Visibility Algorithms*

Gokarna Sharma† Costas Busch† Supratik Mukhopadhyay†


## Abstract

We consider the fundamental MUTUAL VISIBILITY problem for a set of $n$ identical autonomous point robots ($n$ is not known to the robots) that operate following Look-Compute-Move cycles starting from arbitrary distinct positions in the Euclidean plane under obstructed visibility – a robot $r_i$ can see robot $r_j, r_j \neq r_i$, if and only if there is no other robot in the line segment joining their positions. The objective is to determine a schedule to reposition these robots without collisions such that they reach in finite time a configuration where they all see each other. In the recently proposed so-called robots with lights model, Di Luna *et al.* [15] gave two deterministic algorithms Contain and Shrink for this problem; however, no runtime bounds were given except the proof that they terminate in finite time. In this paper, we first study the runtime bounds of these algorithms in the fully synchronous setting showing that Contain is tight ($\Theta(n)$ rounds) and Shrink needs $\Omega(n^2)$ rounds in the worst-case. We then present a new deterministic algorithm, called Modified_Shrink, for fully synchronous setting that solves this problem in $\mathcal{O}(n \log n)$ rounds, improving significantly over Shrink. We also show that Modified_Shrink has the lower bound of $\Omega(n)$ rounds.


## 1 Introduction

Consider a set of $n$ autonomous point robots ($n$ is not known to the robots) in the distinct positions in the Euclidean plane $\mathbb{R}^2$ which are anonymous, indistinguishable, and without any direct means of communication. Each robot is equipped with a local coordinate system and sensor capabilities (i.e., vision) to determine the positions of other robots. The local coordinate system of a robot may be different with that of other robots. The robots execute the same algorithm. They operate in *Look-Compute-Move* cycles, i.e., when a robot becomes active, it uses its vision to get a snapshot of its surroundings (*Look*), computes a destination point based on the snapshot (*Compute*), and finally moves towards the destination (*Move*), if any. Most of the literature assumes that the robots are *oblivious* - each robot has no memory of its past Look-Compute-Move actions - and visibility is *unobstructed* - three collinear robots are assumed to be mutually visible to each other [2, 8, 9, 12, 19, 22].

In this paper, we consider *obstructed visibility* [4, 3, 10, 1, 5, 6] under which a robot $r_i$ can see robot $r_j, r_i \neq r_j$, if and only if there is no other robot in the line segment joining their positions. We study the following fundamental MUTUAL VISIBILITY problem: Starting from the arbitrary distinct positions in the Euclidean plane $\mathbb{R}^2$, determine a schedule to reposition the robots without collisions such that they reach within finite time a configuration where they all see each other. Note that robots moves do not follow grid coordinates of the plane $\mathbb{R}^2$, i.e., we do not assume the existence of some underlying universal grid in $\mathbb{R}^2$. Although obstructed visibility is considered before in the classical oblivious robots model for the SPREADING problem [4] and in the so-called *fat robots* model [1, 6, 12, 17], the technique of [4] cannot be generalized for MUTUAL VISIBILITY, since it works only in the one-dimensional space $\mathbb{R}^1$, and the techniques of [1, 6, 12, 17] are also not suitable, since collisions are allowed and used as an explicit communication tool.

Di Luna *et al.* [16] were the first to study MUTUAL VISIBILITY problem. They studied MUTUAL VISIBILITY in the *robots with lights* model initially suggested by Peleg [18], where each robot has an externally visible persistent light that can assume colors from a fixed set of colors and the color set is identical to all the robots. The robots communicate with other robots using these colored lights [12, 7, 11, 13, 21, 18]; the reason for considering robots with lights model is that there is no MUTUAL VISIBILITY algorithm in the classical oblivious robots model when $n$ is not known. The lights are not erased at the end of each cycle in this model and the robots are oblivious, except the direct communication capability provided by lights. Moreover, this model corresponds to the classical oblivious robots model when the number of colors $c = 1$ in the color set, since a light with only one possible color acts as no light. Di Luna *et al.* [16] gave a deterministic algorithm that solves MUTUAL VISIBILITY with $c = 6$ colors in the semi-synchronous setting and with $c = 10$ colors in the asynchronous setting. Later, Di Luna *et al.* [15] gave two deterministic algorithms Contain and Shrink with $c = 3$ and $c = 2$

*The project is supported by Army Research Office (ARO) under Grant #W911-NF1010495. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the ARO or the United States Government.

†School of Electrical Engineering and Computer Science, Louisiana State University, {gokarna, busch, supratik}@csc.lsu.edu.

colors, respectively, in the semi-synchronous setting.

Di Luna *et al.* [15] proved the correctness of both algorithms Contain and Shrink. However, no runtime bounds were given except the proof that they terminate in finite time. Recently, Vaidyanathan *et al.* [20] gave an algorithm similar to Contain for MUTUAL VISIBILITY in the fully synchronous setting and proved that it has running time of $\mathcal{O}(\log n)$ rounds. However, their algorithm assumes *chirality* [1, 6] and does not avoid robot collisions due to the crossing of paths during robot movements.

In this paper, we consider the fully synchronous setting (where all robots are activated in a round and robots perform their cycles in a perfectly synchronous setting) and study the runtime bounds of MUTUAL VISIBILITY algorithms. In particular, we have made following three contributions.

- We show that Contain [15] has the tight bound of $\Theta(n)$ rounds on running time.
- We show that there exists an initial configuration of $n$ robots in which Shrink [15] needs $\Omega(n^2)$ rounds.
- We present a new deterministic algorithm, called Modified_Shrink, for fully synchronous setting that uses $c = 3$ colors and needs only $\mathcal{O}(n \log n)$ rounds to solve MUTUAL VISIBILITY starting from any initial configuration of $n$ robots. This is a significant improvement over the runtime bound of Shrink which is at least $\Omega(n^2)$ rounds. We also prove that Modified_Shrink has a lower bound of $\Omega(n)$ rounds.

**Paper Organization:** We proceed as follows. We present model in Section 2. We prove bounds for Contain in Section 3. We then prove a lower bound for Shrink in Section 4. In Section 5, we present and analyze Modified_Shrink. Many proofs are omitted due to space constraints.

## 2 Model

We consider a set of $n$ anonymous robots $\mathcal{R} = \{r_1, r_2, \ldots, r_n\}$ operating in the Euclidean plane $\mathbb{R}^2$; $n$ is not assumed to be known. We denote by $p_i(k) \in \mathbb{R}^2$ the position occupied by robot $r_i \in \mathcal{R}$ at time $k$. A robot $r_i$ sees robot $r_j, r_j \neq r_i$, at time $k$ if and only if the line segment $\overline{p_i(k)p_j(k)}$ does not contain any other robot at time $k$. Two robots $r_i$ and $r_j$ are said to *collide* at time $k$ if $p_i(k) = p_j(k)$. If no ambiguity arises, we omit $k$ from $r_i(k)$ and $p_i(k)$, and use $r_i$ to denote both the robot $r_i$ and its position $p_i$. Each robot $r_i$ has its own coordinate system centered in itself and it knows its position with respect to its coordinate system. Moreover, robots have their own unit of distance which may not agree on the unit of measure of other robots. The robots do not agree on the orientation of their coordinate system, i.e., there is no common notion of the clockwise direction.

Each robot $r_i$ is equipped with an externally visible persistent light which can assume any color from a fixed finite set of colors $\mathcal{C}$. The colors in $\mathcal{C}$ are the same for all robots in $\mathcal{R}$. We use variable $r_i.light$ to denote the light of a robot $r_i$. The color of the light of a robot $r$ at time $k$ can be seen by all robots that are visible to $r$ at time $k$. Robots are oblivious − do not remember decisions performed in previous cycle − and a robot's decision at any cycle is only based on the positions of the robots visible to it at that cycle. Robots are *autonomous* (i.e., without any external control), *indistinguishable* (i.e., do not have external markings), and do not have any direct means of communication (except the lights). Moreover, they are *anonymous* (i.e., do not have internal identifiers). Each robot executes the same algorithm locally every time it is activated.

A *configuration* $\mathbb{C}$ is a set of $n$ tuples in $\mathbb{C} \times \mathbb{R}^2$ which defines the position and color of a robot. Let $\mathbb{C}_k$ denotes the configuration at time $k$. Let $\mathbb{C}_k(r_i)$ denotes the configuration $\mathbb{C}_k$ for robot $r_i$. A configuration $\mathbb{C}$ is *obstruction-free* if $\forall r_i \in \mathcal{R}$, we have that $|\mathbb{C}(r_i)| = n$ (i.e., all robots can see each other). Let $\mathbb{H}_k$ denotes the convex hull formed by $\mathbb{C}_k$ which can be easily computed using Graham's convex hull algorithm [14]. Let $\partial\mathbb{H}_k = \mathcal{V}_k \cup \mathcal{E}_k$ denotes the robots in the boundary of $\mathbb{H}_k$, where $\mathcal{V}_k \subseteq \mathcal{R}$ are the set of robots lying at the vertices of $\mathbb{H}_k$ and $\mathcal{E}_k \subseteq \mathcal{R}$ are the set of robots lying at the sides (or edges) of $\mathbb{H}_k$. The robots in the set $\mathcal{V}_k$ are called *vertex robots* and in the set $\mathcal{E}_k$ are called *edge robots*. The robots in $\mathcal{V}_k \cup \mathcal{E}_k$ are called *boundary robots*. The robots in the set $\mathbb{H}_k \backslash \partial\mathbb{H}_k$ are called *internal robots*. Given a robot $r_i \in \mathcal{R}$, we denote by $\mathbb{H}_k(r_i)$ the convex hull of $\mathbb{C}_k(r_i)$. Given two points $a, b \in \mathbb{R}^2$, we denote by $\overleftrightarrow{ab}$ the line that contains them.

We assume that the execution starts at time 0. Therefore, at time $t = 0$, the robots start in an arbitrary configuration $\mathbb{C}_0$ occupying distinct positions in $\mathbb{R}^2$ and the color of the light of each robot is set to *Off*. The MUTUAL VISIBILITY problem is defined as follows: Given any $\mathbb{C}_0$, reach in finite time an obstruction-free configuration without collisions. An algorithm is said to solve MUTUAL VISIBILITY if it always achieves an obstruction-free configuration regardless of the choices of the adversary and from any arbitrary $\mathbb{C}_0$.

We assume that, when active, each robot $r_i \in \mathcal{R}$ performs a sequence of *Look-Compute-Move* (LCM) operations: a robot takes the snapshot of the positions of the robots visible to it in its own coordinate system (*Look*); executes its algorithm using the snapshot which returns a destination point $x \in \mathbb{R}^2$ and a color $c \in \mathcal{C}$ (*Compute*); and sets its own light to color $c$ and moves towards the computed destination $x \in \mathbb{R}^2$ (if $x$ is differ-

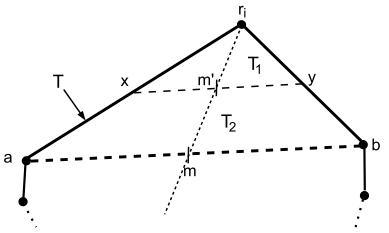


Figure 1: An illustration of $T$, $T_1$, and $T_2$ of a vertex robot $r_i$ in $\mathbb{H}(r_i)$, where $a$ and $b$, respectively, are the neighbor robots of $r_i$ in $\mathbb{H}(r_i)$ in its counterclockwise and clockwise direction, $x$, $y$, and $m$ are the midpoints of line segments $\overline{r_i a}$, $\overline{r_i b}$, and $\overline{ab}$, respectively, and $m'$ is the intersection point of $\overline{r_i m}$ and $\overline{xy}$.

ent than its current position), if any (*Move*). We assume the rigid moves throughout the paper in the sense that the movement of robots are not controlled by an adversary and every robot reaches its destination at all times when it moves from a current position to its computed destination. Moreover, we consider a fully synchronous scheduler for the activation of the robots in $\mathcal{R}$. In the fully synchronous scheduler, the time is discrete and at each time instant $k$ all the robots of $\mathcal{R}$ are activated and perform their LCM operations instantaneously, ending at time $k+1$. Therefore, we use round $k$ instead of time $k$ from now on. Finally, we measure the quality of the algorithm by counting the number of rounds until the robots have reached the mutual visibility configuration (all robots are the vertices of $\mathbb{H}$).

As shown in Fig. 1, let $r_i$ be a vertex of $\mathbb{H}$ and $a$ and $b$ are its counterclockwise and clockwise neighbors in $\mathbb{H}$. Moreover, let $x$ and $y$ be the midpoints of line segments $\overline{r_i a}$ and $\overline{r_i b}$, respectively, and $m$ be the midpoint of line segment $\overline{ab}$. We have that, according to construction, $\overline{xy}$ is parallel to $\overline{ab}$. For each vertex robot $r_i$, we denote by $T$ the triangular area $r_i ab$, by $T_1$ the triangular area $r_i xy$, and by $T_2$ the trapezoidal area $xyba$ (i.e., $T_2 := T \backslash T_1$). When we say that a robot $w$ is *closest* to $r_i$ then we mean that there is no other robot in the area of $\mathbb{H}(r_i)$ between $r_i$ and a line parallel to $\overline{ab}$ (or $\overline{xy}$) that passes through $w$'s position.

## 3 Tight Bounds for Contain Algorithm

Contain [15] has two phases: an *interior depletion* phase and a *vertex adjustment* phase. The second phase is executed only after the first phase is finished. In the interior depletion phase, the robots in the interior of $\mathbb{H}$ move towards the boundary of $\mathbb{H}$ and in the vertex adjustment phase the robots in the vertices of $\mathbb{H}$ move towards the interior of $\mathbb{H}$ to reach a strictly convex configuration with all the robots being in the vertices of $\mathbb{H}$. Three colors are used, namely $\mathcal{C} = \{Off, External, Adjusting\}$.

We prove the following lemma for the lower bound.

**Lemma 1** *There is an initial configuration $\mathbb{C}_0$ of the robots in which* Contain *takes $\Omega(n)$ rounds to solve* Mutual Visibility *in the fully synchronous setting.*

We prove the following lemma for the upper bound.

**Lemma 2** *Starting from any initial configuration of a set of n robots,* Contain *needs $\mathcal{O}(n)$ rounds to solve* Mutual Visibility *in the fully synchronous setting.*

Combining the lower and upper bounds of Lemmas 1 and 2, we obtain the following theorem.

**Theorem 3** *The round complexity of* Contain *for* Mutual Visibility *is $\Theta(n)$ in the worst-case in the fully synchronous setting.*

## 4 Lower Bound for Shrink Algorithm

Shrink works as follows. The vertex robots set their light to *Vertex*. Let $r_i$ be a vertex robot of $\mathbb{H}(r_i)$ and $a$ and $b$ be $r_i$'s counterclockwise and clockwise neighbors both in the boundary of $\mathbb{H}(r_i)$. Let $x$ and $y$ be the midpoints of line segments $\overline{r_i a}$ and $\overline{r_i b}$, respectively, and $m$ be the midpoint of $\overline{ab}$. If there is an interior robot, say $r'$, in $T_1$, then $r_i$ moves to some point in $T_1$ in the line segment that is parallel to $\overline{xy}$ and passes through $r$. If there are more than one robot in $T_1$, then some point in the line segment parallel to $\overline{xy}$ passing through the closest robot is chosen. However, if there is no robot inside $T_1$ (i.e. all interior robots are outside $T_1$), then $r_i$ moves to the point $m'$ in $\overline{xy}$, irrespective of the positions of the interior robots, where $m'$ is the point in which the line segment $\overline{r_i m}$ intersects $\overline{xy}$ (see Fig. 1). If there is only one robot in the interior of $\mathbb{H}(r_i)$, then that interior robot moves to the midpoint of some edge in the boundary of $\mathbb{H}(r_i)$. Two colors are used, namely $\mathcal{C} = \{Off, Vertex\}$. We prove the following theorem for the lower bound.

**Theorem 4** *There is an initial configuration $\mathbb{C}_0$ of the $n$ robots in which* Shrink *takes $\Omega(n^2)$ rounds to solve* Mutual Visibility *in the fully synchronous setting.*

## 5 The Modified_Shrink Algorithm

We present Modified_Shrink which improves significantly over the runtime of Shrink in the fully synchronous setting. The pseudocode of Modified_Shrink is given in Algorithms 1. Algorithm 1 uses Algorithms 2–4 as subroutines to accomplish the mutual visibility of robots starting from any arbitrary initial configuration $\mathbb{C}_0$.

Modified_Shrink uses three colors in the set $\mathcal{C}$ of colors, namely $\mathcal{C} = \{Off, Vertex, Edge\}$. The colors in the set $\mathcal{C}$ are used by robots to detect whether Mutual

**Algorithm 1:** Modified_Shrink algorithm for any round $k > 0$

1 // Look-Compute-Move cycle for each robot $r_i \in \mathcal{R}$
2 $\mathbb{C}_k(r_i) \leftarrow$ configuration $\mathbb{C}_k$ for robot $r_i$ (including $r_i$);
3 $\mathbb{H}_k(r_i) \leftarrow$ convex hull of the positions of the robots in $\mathbb{C}_k(r_i)$;
4 **if** $|\mathbb{C}_k(r_i)| = 3 \wedge \mathbb{H}_k(r_i)$ *is a line segment* **then**
5  Move orthogonal to (the line segment) $\mathbb{H}_k(r_i)$ by any non-zero distance;
6 **else**
7  **if** $r_i$ *is in vertex of* $\mathbb{H}_k(r_i)$ **then** $Corner(r_i, \mathbb{C}_k(r_i), \mathbb{H}_k(r_i))$;
8  **else if** $r_i$ *is in edge of* $\mathbb{H}_k(r_i)$ **then** $Side(r_i, \mathbb{C}_k(r_i), \mathbb{H}_k(r_i))$;
9  **else if** $\forall r \in \mathbb{C}_k(r_i)\backslash\{r_i\}$, $r.light \in \{Vertex, Edge\} \wedge r_i$ *is in interior of* $\mathbb{H}_k(r_i)$ **then** $Interior(r_i, \mathbb{H}_k(r_i))$;

**Algorithm 2:** $Corner(r_i, \mathbb{C}_k(r_i), \mathbb{H}_k(r_i))$

1 **if** $r_i.light = Off$ **then** $r_i.light \leftarrow Vertex$;
2 **if** $\forall r \in \mathbb{C}_k(r_i)$, $r.light = Vertex$ **then** Terminate;
3 **else if** $|\mathbb{C}_k(r_i)| > 2$ **then**
4  $a \leftarrow$ counterclockwise neighbor on the boundary of $\mathbb{H}_k(r_i)$;
5  $b \leftarrow$ clockwise neighbor on the boundary of $\mathbb{H}_k(r_i)$;
6  $x \leftarrow$ midpoint of the line segment $\overline{r_i a}$;
7  $y \leftarrow$ midpoint of the line segment $\overline{r_i b}$;
8  **if** *there exists at least a robot in* $\mathbb{C}_k(r_i)\backslash\{r_i\}$ *with light Off* **then**
9  $r' \leftarrow$ robot in $\mathbb{C}_k(r_i)\backslash\{r_i\}$ with light *Off* that is closest to $r_i$ w.r.t. the line parallel to the line segment $\overline{ab}$ (if more than one robot satisfies this criteria, choose as $r'$ the robot that is closer to $b$);
10  **if** $r'$ *is not in the triangular area* $r_iab$ **then**
11  Move to the midpoint of the line segment $\overline{r_i r'}$;
12  **else if** $r'$ *is in the triangular area* $r_iab \wedge r'$ *is not in the triangular area* $r_ixy$ **then**
13  Move to the intersection point of the line segments $\overline{r_i r'}$ and $\overline{xy}$;
14  **else if** $r'$ *is in the triangular area* $r_ixy$ **then**
15  $L \leftarrow$ line parallel to $\overline{ab}$ that passes through $r'$;
16  $z \leftarrow$ intersection point of $L$ and $\overline{r_i b}$;
17  Move to the midpoint of the line segment $\overline{r'z}$;
18  **else if** *there exists at least a robot in* $\mathbb{C}_k(r_i)\backslash\{r_i\}$ *with light Edge* **then**
19  Move to the midpoint of the line segment $\overline{xy}$;

Visibility is solved and correctly terminate their computation. The lights of all robots in $\mathcal{R}$ are set to *Off* in the initial configuration $\mathbb{C}_0$. When a robot after activation in some round $k > 0$ realizes that it is a vertex of $\mathbb{H}$, it sets its light to *Vertex* (Line 7 of Algorithm 1, Line 1 of Algorithm 2). This task is easy since, if a robot $r_i$ with light *Off* after activation in some round $k > 0$ sees that $\mathbb{C}_k(r_i)$ contains a region of plane that is free of robots and wider than 180°, then $r_i$ knows it is a vertex of $\mathbb{H}$. If a robot realizes after activation in some round $k > 0$ that it is on an edge of $\mathbb{H}$, it sets its light to *Edge* (Line 8 of Algorithm 1, Line 1 of Algorithm 3). Similar to the realization of vertex robots, if a robot $r_i$ with light *Off* after activation in some round $k$ sees that $\mathbb{C}_k(r_i)$ contains a region of plane that is free of robots and wide exactly 180°, then $r_i$ knows it is on an edge of $\mathbb{H}$. When the lights of all the robots that are seen by a robot are set to *Vertex*, the robot knows that it can see all the robots in $\mathcal{R}$ and hence it terminates (without the knowledge of $n$, the total number of robots).

Since our algorithm uses the convex hull shrinking process, the vertex and edge robots move inside. We now describe how vertex robots move and give the details on how edge robots move in the next paragraph. If there is an interior robot (i.e., a robot with light *Off*), say $r'$, in $T_1$, then the vertex robot $r_i$ moves somewhere in the line parallel to the line segment $\overline{ab}$ that passes through $r'$, where $a$ and $b$ are the neighbor robots of $r_i$ in $\mathbb{H}_k(r_i)$ in its counterclockwise and clockwise direction, respectively. If there is more than one robot in $T_1$, the closest one from $r_i$ (w.r.t. a line parallel to $\overline{ab}$) is chosen as the robot $r'$ (Lines 14-17 of Algorithm 2). In case all the interior robots are outside $T_1$ then if there are robots in $T_2$, $r_i$ moves to the point in the line segment $\overline{xy}$ that intersects the line segment $\overline{r_i r'}$, where $r'$ is the robot in $T_2$ that is closest to $r_i$ again w.r.t. a line parallel to $\overline{ab}$, and $x$ and $y$, respectively, are the midpoints of the line segments $\overline{r_i a}$ and $\overline{r_i b}$ (Lines 12,13 of Algorithm

**Algorithm 3:** $Side(r_i, \mathbb{C}_k(r_i), \mathbb{H}_k(r_i))$

1 **if** $r_i.light = Off$ **then** $r_i.light \leftarrow Edge$;
2 **if** *there exists at least a robot in* $\mathbb{C}_k(r_i)\backslash\{r_i\}$ *with light Off* **then**
3 $a \leftarrow$ counterclockwise neighbor on the boundary of $\mathbb{H}_k(r_i)$;
4 $b \leftarrow$ clockwise neighbor on the boundary of $\mathbb{H}_k(r_i)$;
5 $r_j \leftarrow$ closest robot to $r_i$ in $\mathbb{C}_k(r_i)\backslash\{r_i\}$ with light *Off* w.r.t. a line parallel to line segment $\overline{ab}$;
6 Move to the midpoint of the line segment $\overline{r_i r_j}$;

**Algorithm 4:** $Interior(r_i, \mathbb{H}_k(r_i))$

1 **if** $r_i$ *is not in the triangular area formed by any three consecutive robots of* $\mathbb{H}_k(r_i)$ **then**
2 $x \leftarrow$ midpoint of any edge between two consecutive robots of $\mathbb{H}_k(r_i)$;
3 Move to the midpoint of the line segment $\overline{r_i x}$;
4 **else**
5 $e \leftarrow$ the closest among two edges of the triangular area that are in $\mathbb{H}_k(r_i)$;
6 Move to the midpoint of the edge $e$;

2). When all the interior robots are outside $T$, $r_i$ moves halfway to the line segment $overliner_i r'$ connecting $r_i$ with the robot $r'$ that is closest to it (Lines 10,11 of Algorithm 2). Note that if more than one robot is closest to the vertex robot $r_i$ according to our criteria (w.r.t. a line parallel to $\overline{ab}$), then the robot that is closer to $b$ among the closest robots is chosen as $r'$ (Line 9 of Algorithm 2).

On the other hand, a robot, $r_j$, on the edge of $\mathbb{H}$ (which is not a vertex of $\mathbb{H}$) moves halfway to an internal robot that is closest to $r_j$ w.r.t. a line parallel to $\overline{ab}$ ($\overline{ab}$ in this case is in fact a straight line segment that passes through $r_j$) (Lines 2–5 of Algorithm 3).

Moreover, there can be a situation in our algorithm that there is no robot in the interior of $\mathbb{H}$ but there are still some robots on the edges of $\mathbb{H}$. The robots in the edges do not move since there should not be any robot with light *Off* in the system for this situation to happen. On the other hand, the vertex robots recognize this situation and start moving to the midpoint of the line segment $\overline{xy}$ (Lines 18,19 of Algorithm 2). These moves of vertex robots are sufficient since we can show that eventually all edge robots become vertices even under these moves.

We now have two special cases in our algorithm. The first special case is when there is only one internal robot. In this case, we can show that if the internal robot does not move, Mutual Visibility cannot be achieved without collisions since, all the robots in $\mathcal{R}$ converge to the position of the only internal robot. However, our algorithm resolves this situation as follows. When there is only one robot, say $w$, in the interior of $\mathbb{H}_k(w)$, then the robot $w$ recognizes this situation and moves towards the boundary of $\mathbb{H}_k(w)$. This recognition is easy as all the robots $w$ sees have lights either *Vertex* or *Edge* (Line 9 of Algorithm 1). If $w$ is inside the triangular area $r_i ab$ of some vertex robot $r_i$, it chooses the closest edge between $r_i a$ and $r_i b$ and moves to the midpoint of that edge (Lines 1, 5, 6 of Algorithm 4). Otherwise, it moves halfway from its location to the line segment connecting it with the midpoint of any edge between two consecutive robots of $\mathbb{H}_k(w)$ (Lines 2,3 of Algorithm 4). The second special case is when a robot $r_i \in \mathcal{R}$ sees only two other robots (Lines 4,5 of Algorithm 1). In this case, $H_k(r_i)$ must be a line segment. The robot $r_i$ then moves orthogonal to $H_k(r_i)$ by any non-zero distance. These movements translate line segment $H_k(r_i)$ into a polygonal $H_k(r_i)$ which remains as polygonal $H_k(r_i)$ in future rounds.

### 5.1 Analysis of the Modified_Shrink Algorithm

We here analyze Modified_Shrink for both correctness and running time. We first show that the paths that robots follow when they move inside do not cross which is essential to show that Modified_Shrink avoids collisions due to hitting each other while relocating. We have the following lemma.

**Lemma 5** *The paths of robots do not cross during the execution of* Modified_Shrink.

We now show that two robots do not land up to the same position during the execution of Modified_Shrink which is essential to prove that there is no collision due to position sharing. Note that in $\mathbb{C}_0$, robots do not share their positions since it is assumed that they start from the distinct positions (otherwise no collision requirement can not be achieved).

**Lemma 6** *Two robots do not land up to the same position during the execution of* Modified_Shrink.

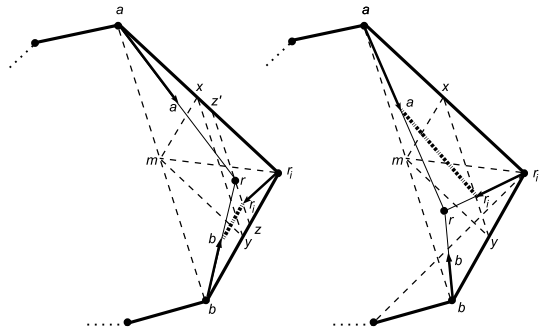


Figure 2: An illustration of robot movements in Modified_Shrink: (left) the closest robot to $r_i$, $r$, is in $T_1$; (right) the closest robot to $r_i$, $r$, is in $T_2$.

Combining Lemmas 5 and 6, we obtain the following theorem on collision avoidance property of Modified_Shrink.

**Theorem 7** *Starting from any configuration of $n$ robots,* Modified_Shrink *avoids robot collisions.*

We now show that vertex robots remain vertex which is essential to prove the convergence property of our algorithm and guarantee progress towards a mutual visibility configuration.

**Lemma 8** *No vertex robot of $\mathbb{H}$ becomes internal or edge robot during the execution of* Modified_Shrink.

**Proof.** When there is no interior robot in the triangular area $T$ of the vertex robots, the external (vertex and edge) robots converge to the same limit, i.e., in every round, all the robots in the boundary of $\mathbb{H}$ move exactly half distance to their closest robots in the line segment connecting them to their closest internal robots in $\mathbb{H}$. It can be easily seen that this process guarantees that the vertices of $\mathbb{H}$ remain as vertices of $\mathbb{H}$ in every future round. Therefore, we focus on the scenario where some vertex robots have internal robots in their triangular areas $T$ and others do not have internal robots in their triangular areas.

We will show that the moves of some vertex robots to the positions inside $T_1$ and the moves of other vertex robots to the positions outside $T_1$ still guarantee that vertex robots remain as vertices of $\mathbb{H}$. One example configuration for such scenario is given in Fig. 2. Let $r_i$ be a vertex robot in $\mathbb{H}$ and $a$ and $b$ are its neighbors in $\mathbb{H}$ with $a$ being in its counterclockwise direction and $b$ being in its clockwise direction. Let $r$ be the robot that is inside the triangular area $T$ of $r_i$. Assume that $r$ is also the closest robot in the interior of $\mathbb{H}$ from the vertex robots $a$ and $b$ and it is not inside the triangular area $T$ of both $a$ and $b$. According to Algorithm 1, if $r$ is inside the triangular area $T_1$ of $r_i$, $r_i$ moves as shown in the left of Fig. 2 in the line, say $L$, that is parallel to the line segment $\overline{xy}$ that passes through the position of $r$. The exact point where $r_i$ moves is the midpoint of the line segment $\overline{rz}$ of the line $L$, where $x$ is the intersection point of the line $L$ and the line segment $\overline{r_ib}$. The robots $a$ and $b$ move halfway to $r$ since $r$ is outside the triangular area $T$ of both $a$ and $b$.

We now show that the convexity of $\mathbb{H}$ is maintained in this situation. We first consider robot $b$ and then the argue for robot $a$. We have that line segments $\overline{br}$ and $\overline{r_ib}$ intersect at $b$ before $b$ and $r_i$ move inside which is also the vertex of $\mathbb{H}$. After they move inside, the line segment $\overline{r_ib}$ (bold dotted line in the left of Fig. 2) connecting the new positions of $r_i$ and $b$ is parallel to the line segment $\overline{r_ib}$ connecting their old positions because their new positions are the midpoints of two sides $rb$ and $rz$ of the triangle $rbz$. Now since $r$ is also the closest robot to $a$ the move of $a$ makes the line segment $\overline{ar}$ (from $a$'s new position to $r$) parallel to the line segment $\overline{ar_i}$ (from $a$'s old position to $r_i$) if $r_i$ moves to the midpoint of the line segment $\overline{rz'}$ (the argument here is similar to the one used for robot $b$), otherwise $\overline{ar}$ remains as the segment of line $\overleftrightarrow{ar}$ that intersects $\overleftrightarrow{r_ia}$ at $a$. Therefore, under any movements of $r_i, a$, and $b$, $r_i$ does not become internal or edge robot because neither $a$ nor $b$ crosses the line segment $\overline{zz'}$ to reach some point in the triangular area $r_izz'$ of $r_i$. Note that $r$ actually becomes vertex after the moves of $r_i$, $a$, and $b$. Since this process is applied by all the vertex robots of $\mathbb{H}$, it is clear that the convexity of $\mathbb{H}$ is maintained and no vertex robot becomes an internal or edge robot.

Consider now the scenario where $r$ is inside the trapezoidal area $T_2$ of $r_i$ (the right of Fig. 2). In this case, $r_i$ moves to the point where line segments $\overline{xy}$ and $\overline{r_ir}$ intersect. As the distance from $r_i$ to its new position (after move) is at least half of $\overline{r_ir}$, the line segments connecting the new positions of $a, r_i$, and $r_i, b$ become parallel to the current edges of $\mathbb{H}$, $ap$ and $pb$, when $r$ is at some position in the line segment $\overline{ab}$. Therefore, under any movements of $r_i, a$, and $b$, $r_i$ does not become internal or edge robot because neither $a$ nor $b$ crosses the line segment $\overline{xy}$ to reach some point in the triangular area $T_1$ of $r_i$. Hence, combining the above claims, the lemma follows. □

We are now ready to analyze the runtime bound of Modified_Shrink.

**Theorem 9** Modified_Shrink *solves* Mutual Visibility *in $O(n \log n)$ rounds using lights with 3 colors in the fully synchronous setting.*

**Proof.** When $\mathbb{H}$ is a line in $\mathbb{C}_0$ it becomes a polygonal $\mathbb{H}$ in one round due to the fully synchronous setting and it is easy to see that once line $\mathbb{H}$ is transitioned to polygonal $\mathbb{H}$, it does not become line $\mathbb{H}$ again in future. Starting from polygonal $\mathbb{H}$, according to Algorithm 1,

when there exists a robot in $T_1$, then it becomes vertex in one round. When there is a robot in $T_2$, then the robot reaches at least halfway close to it in next round. Therefore, the worst-case number of rounds of Algorithm 1 is when all the interior robots are not inside any $T$ of vertex robots. However, we have from Algorithm 1 that external robots reach halfway to those interior robots (even if they are not inside $T$ of any vertex robots) in every round. As vertex robots remain vertex (Lemma 8) and external robots move halfway to the interior robots in each round, after at most $\mathcal{O}(\log n)$ rounds, at least one internal robot becomes an external robot (vertex or edge). This is because the distance between a vertex robot and its closest internal robot decreases by half in every round and the closest internal robot for a vertex robot remains as closest until it eventually becomes an external robot. Therefore, as there are $n$ robots in $\mathcal{R}$, they become external in at most $\mathcal{O}(n \log n)$ rounds. Moreover, we need at most $\mathcal{O}(n)$ rounds to make robots in edges the vertex robots after all internal robots reach the boundary of $\mathbb{H}$ (Lines 18,19 of Algorithm 2). Therefore, Algorithm 1 needs at most $\mathcal{O}(n \log n) + \mathcal{O}(n) = \mathcal{O}(n \log n)$ rounds. □

We have the following theorem for the lower bound of Modified_Shrink which shows the inherent difficulty in obtaining faster algorithms for the MUTUAL VISIBILITY problem.

**Theorem 10** *There exists an initial configuration $\mathbb{C}_0$ of the robots in which* Modified_Shrink *takes $\Omega(n)$ rounds to solve* MUTUAL VISIBILITY *in the fully synchronous setting.*

## References


[1] C. Agathangelou, C. Georgiou, and M. Mavronicolas. A distributed algorithm for gathering many fat mobile robots in the plane. In *PODC*, pages 250–259, 2013.

[2] N. Agmon and D. Peleg. Fault-tolerant gathering algorithms for autonomous mobile robots. In *SODA*, pages 1070–1078, 2004.

[3] K. Bolla, T. Kovacs, and G. Fazekas. Gathering of fat robots with limited visibility and without global navigation. In *SIDE*, pages 30–38, 2012.

[4] R. Cohen and D. Peleg. Local spreading algorithms for autonomous robot systems. *Theor. Comput. Sci.*, 399(1-2):71–82, June 2008.

[5] A. Cord-Landwehr, B. Degener, M. Fischer, M. Hüllmann, B. Kempkes, A. Klaas, P. Kling, S. Kurras, M. Märtens, F. Meyer auf der Heide, C. Raupach, K. Swierkot, D. Warner, C. Weddemann, and D. Wonisch. Collisionless gathering of robots with an extent. In *SOFSEM*, pages 178–189, 2011.

[6] J. Czyzowicz, L. Gasieniec, and A. Pelc. Gathering few fat mobile robots in the plane. *Theor. Comput. Sci.*, 410(6-7):481–499, 2009.

[7] S. Das, P. Flocchini, G. Prencipe, N. Santoro, and M. Yamashita. The power of lights: Synchronizing asynchronous robots using visible bits. In *ICDCS*, pages 506–515, 2012.

[8] X. Défago and S. Souissi. Non-uniform circle formation algorithm for oblivious mobile robots with convergence toward uniformity. *Theor. Comput. Sci.*, 396(1-3):97–112, 2008.

[9] Y. Dieudonné, O. Labbani-Igbida, and F. Petit. Circle formation of weak mobile robots. *TAAS*, 3(4), 2008.

[10] A. Dutta, S. G. Chaudhuri, S. Datta, and K. Mukhopadhyaya. Circle formation by asynchronous fat robots with limited visibility. In *ICDCIT*, pages 83–93, 2012.

[11] A. Efrima and D. Peleg. Distributed models and algorithms for mobile robot systems. In *SOFSEM*, pages 70–87, 2007.

[12] P. Flocchini, G. Prencipe, and N. Santoro. Distributed computing by oblivious mobile robots. *Synthesis Lectures on Distributed Computing Theory*, 3(2):1–185, 2012.

[13] P. Flocchini, N. Santoro, G. Viglietta, and M. Yamashita. Rendezvous of two robots with constant memory. In *SIROCCO*, pages 189–200, 2013.

[14] R. L. Graham. An efficient algorithm for determining the convex hull of a finite planar set. *Inf. Process. Lett.*, 1(4):132–133, 1972.

[15] G. A. D. Luna, P. Flocchini, S. G. Chaudhuri, F. Poloni, N. Santoro, and G. Viglietta. Mutual visibility by luminous robots without collisions. To appear in Information and Computation, *Available at* `http://arxiv.org/abs/1503.04347`, 2015.

[16] G. A. D. Luna, P. Flocchini, S. G. Chaudhuri, N. Santoro, and G. Viglietta. Robots with lights: Overcoming obstructed visibility without colliding. In *SSS*, pages 150–164, 2014.

[17] G. A. D. Luna, P. Flocchini, F. Poloni, N. Santoro, and G. Viglietta. The mutual visibility problem for oblivious robots. In *CCCG*, 2014.

[18] D. Peleg. Distributed coordination algorithms for mobile robot swarms: New directions and challenges. In *IWDC*, pages 1–12, 2005.

[19] I. Suzuki and M. Yamashita. Distributed anonymous mobile robots: Formation of geometric patterns. *SIAM J. Comput.*, 28(4):1347–1363, 1999.

[20] R. Vaidyanathan, C. Busch, J. L. Trahan, G. Sharma, and S. Rai. Logarithmic-time complete visibility for robots with lights. In *IPDPS*, pages 375–384, 2015.

[21] G. Viglietta. Rendezvous of two robots with visible bits. In *ALGOSENSORS*, pages 291–306, 2013.

[22] M. Yamashita and I. Suzuki. Characterizing geometric patterns formable by oblivious anonymous mobile robots. *Theor. Comput. Sci.*, 411(26-28):2433–2453, 2010.