Rendezvous of Asynchronous Mobile Robots with Lights

Takashi Okumura[1], Koichi Wada[2(✉)], and Yoshiaki Katayama[3]

[1] Graduate School of Science and Engineering, Hosei University, Tokyo 184-8584, Japan
takashi.okumura.4e@stu.hosei.ac.jp

[2] Faculty of Science and Engineering, Hosei University, Tokyo 184-8485, Japan
wada@hosei.ac.jp

[3] Graduate School of Engineering, Nagoya Institute of Technology, Nagoya 466-8555, Japan
katayama@nitech.ac.jp

Abstract. We study a *Rendezvous* problem for 2 autonomous mobile robots in asynchronous settings with persistent memory called *light*. It is well known that Rendezvous is impossible when robots have no lights in basic common models, even if the system is semi-synchronous. On the other hand, Rendezvous is possible if robots have lights with a constant number of colors in several types of lights [9,21]. In asynchronous settings, Rendezvous can be solved by robots with 3 colors of lights in non-rigid movement and with 2 colors of lights in rigid movement, respectively [21], if the robots can use not only their own light but also the other robot's light (*full-light*), where non-rigid movement means robots may be stopped before reaching the computed destination but can move a minimum distance $\delta > 0$, and rigid movement means robots can reach the computed destination. In semi-synchronous settings, Rendezvous can be solved with 2 colors of full-lights in non-rigid movement.

In this paper, we show that in asynchronous settings, Rendezvous can be solved with 2 colors of full-lights in non-rigid movement if robots know the value of the minimum distance δ. We also show that Rendezvous can be solved with 2 colors of full-lights in general non-rigid movement if we consider some reasonable restricted class of asynchronous settings.

1 Introduction

Background and Motivation

The computational issues of autonomous mobile robots have been research object in distributed computing fields. In particular, a large amount of work has been dedicated to the research of theoretical models of autonomous mobile robots [1–3,6,12,15,18,19]. In the basic common setting, a robot is modeled as a point in a two-dimensional plane and its capability is quite weak. We usually assume that robots are *oblivious* (no memory to record past history), *anonymous* and *uniform* (robots have no IDs and run identical algorithms) [8]. Robots operate in

H.-J. Böckenhauer et al. (Eds.): Hromkovič Festschrift, LNCS 11011, pp. 434–448, 2018.
https://doi.org/10.1007/978-3-319-98355-4_25

Look-Compute-Move (LCM) cycles in this model. In the Look operation, robots obtain a snapshot of the environment and they execute the same algorithm with the snapshot as an input to the Compute operation, and move towards the computed destination in the Move operation. Repeating these cycles, all robots perform a given task. It is difficult for these very weak robot systems to complete the task. Revealing the weakest capability of robots sufficient to attain a given task is one of the most interesting challenges in the theoretical research of autonomous mobile robots.

The problem considered in this paper is *Gathering*, which is one of the most fundamental tasks of autonomous mobile robots. Gathering is the process of n mobile robots, initially located on arbitrary positions, to meet within finite time at a location, not known a priori. When there are two robots in this setting, this task is called *Rendezvous*. In this paper, we focus on Rendezvous in asynchronous settings and we reveal the weakest additional assumptions for this problem.

Since Gathering and Rendezvous are simple but essential problems, they have been intensively studied and a number of possibility and/or impossibility results have been shown under different assumptions [1–3,5–7,10,13–18]. The solvability of Gathering and Rendezvous depends on the activation schedule and the synchronization level. Usually three basic types of schedulers are identified, the fully synchronous (FSYNC), the semi-synchronous (SSYNC) and the asynchronous (ASYNC). In the FSYNC model, there are common rounds and in each round all robots are activated simultaneously and Compute and Move are done instantaneously. The SSYNC is the same as FSYNC except that in each round only a subset of the robots is activated. In the ASYNC scheduler, there is no restriction on the notion of time, Compute and Move in each cycle can take an unpredictable amount of time, and the time interval between successive activations is also unpredictable (but these times must be finite). Gathering and Rendezvous are trivially solvable in FSYNC by using an algorithm that moves to the center of gravity. However, Rendezvous cannot be solved even in SSYNC without any additional assumptions [8].

In [4], persistent memory called *light* has been introduced to reveal the relationship between ASYNC and SSYNC and they show that asynchronous robots with lights equipped with a constant number of colors are strictly more powerful than semi-synchronous robots without lights. In order to solve Rendezvous without any other additional assumptions, robots with lights have been introduced [4,9,21].

Table 1 shows results on solving Rendezvous by robots with lights with each scheduler and movement restriction. In the table, *full-light* means that robots can see not only the lights of other robots but also their own light. In the movement restriction, Rigid means that robots can reach the computed destination. In Non-Rigid, robots may be stopped before reaching the computed destination but move a minimum distance $\delta > 0$. Non-Rigid($+\delta$) means the setting is Non-Rigid and the robots know the value δ. Gathering of robots with lights is discussed in [20].

Table 1. Rendezvous algorithms by robots with full-lights.

Scheduler	Movement	Full-light [21]	No-light [8,17]
FSYNC	Non-rigid	–	◯
SSYNC	Non-rigid	2	×
	Rigid	–	
	Non-rigid(+δ)	–	
ASYNC	Non-rigid	3	×
	Rigid	2	
	Non-rigid(+δ)	?	

◯ and × mean solvable and unsolvable, respectively,
– indicates that this part has been solved under a weaker condition, and
? means this part is not solved.

Our Contribution

In this paper, we consider whether we can solve Rendezvous in ASYNC with the optimal number of colors of light. In SSYNC, Rendezvous cannot be solved with one color but can be solved with 2 colors in Non-Rigid and full-light. On the other hand, Rendezvous in ASYNC can be solved with 3 colors in Non-Rigid and full-light, and with 2 colors in Rigid.

In this paper, we show that Rendezvous in ASYNC can be solved with 2 colors in Non-Rigid(+δ) and full-light. We give a basic Rendezvous algorithm with 2 colors of full-lights (A and B)[1] and it can solve Rendezvous in ASYNC and Rigid, and its variant can also solve Rendezvous in ASYNC and Non-Rigid(+δ). These two algorithms can behave correctly if the initial color of each robot is A. However if the initial color of each robot is B, the algorithm cannot solve Rendezvous in ASYNC and Rigid. It is still open whether Rendezvous can be solved with 2 colors in ASYNC and Non-Rigid. However, we introduce some restricted class of ASYNC called *LC-atomic* ASYNC, and we show that our basic algorithm can solve Rendezvous in this scheduler and Non-Rigid with arbitrary initial color. Here, LC-atomic ASYNC means that we consider the interval from the beginning of each Look operation to the end of the corresponding Compute operation as an atomic one, that is, no robot can observe between the beginning of each Look operation and the end of each Compute operation in every cycle. This is a reasonable sufficient condition so that Rendezvous can be solved with the optimal number of colors of light in ASYNC and No-Rigid.

2 Model and Preliminaries

We consider a set of n anonymous mobile robots $\mathcal{R} = \{r_1, \ldots, r_n\}$ located in $\mathbb{R}^2$. Each robot r_i has a persistent state $\ell(r_i)$ called *light* which may be taken from a finite set of colors L.

[1] This is essentially the same as that in [21].

We denote by $\ell(r_i, t)$ the color of light the robot r_i has at time t and by $p(r_i, t) \in \mathbb{R}^2$ the position occupied by r_i at time t represented in some global coordinate system. Given two points $p, q \in \mathbb{R}^2$, $dis(p, q)$ denotes the distance between p and q.

Each robot r_i has its own coordinate system where r_i is located at its origin at any time. These coordinate systems do not necessarily agree with those of other robots. This means that there is no common unit of distance and no common knowledge of directions of its coordinates and clockwise orientation (*chirality*).

At any point in time, a robot can be active or inactive. When a robot r_i is activated, it executes *Look*-*Compute*-*Move* operations:

– **Look:** The robot r_i activates its sensors to obtain a snapshot which consists of pairs of a light and a position for every robot with respect to its own coordinate system. Since the result of this operation is a snapshot of the positions of all robots, the robot does not notice the movement, even if the robot sees other moving robots. We assume robots can observe all other robots (unlimited visibility).
– **Compute:** The robot r_i executes its algorithm using the snapshot and its own color of light (if it can be utilized) and returns a destination point des_i with respect to its coordinate system and a light $\ell_i \in L$ to which its own color is set.
– **Move:** The robot r_i moves to the computed destination des_i. A robot r is said to *collide* with robot s at time t if $p(r, t) = p(s, t)$ and r is performing *Move* at time t, and r's collision is *accidental* if its destination is not $p(r, t)$. Since robots are seen as points, we assume that accidental collisions are immaterial. A moving robot, upon causing an accidental collision, proceeds in its movement without changes, in a "hit-and-run" fashion [8]. The robot may be stopped by an adversary before reaching the computed destination.[2] If stopped before reaching its destination, a robot moves at least a minimum distance $\delta > 0$. Note that without this assumption an adversary could make it impossible for any robot to ever reach its destination, following a classical Zenonian argument [8]. If the distance to the destination is at most δ, the robot can reach it. In this case, the movement is called *Non-Rigid*. Otherwise, it is called *Rigid*. If the movement is Non-Rigid and the robots know the value of δ, it is called *Non-Rigid(+δ)*.

A scheduler decides which subset of robots is activated for every configuration. The schedulers we consider are asynchronous or semi-synchronous and it is assumed that for every robot r and time t, there exists a time $t'(\geq t)$ at which r becomes active. Usually this scheduler is said to be *fair* [8].

– **ASYNC:** The asynchronous (ASYNC) scheduler activates the robots independently, and the duration of each Compute, Move and the time between successive activities is finite and unpredictable. As a result, a robot can be seen while moving and the snapshot and its actual configuration are not the same, and so its computation may be done with the old configuration.

[2] E.g., due to limits to its motion energy.

- **SSYNC:** The semi-synchronous (SSYNC) scheduler activates a subset of all robots synchronously and their Look-Compute-Move cycles are performed at the same time. We can assume that activated robots at the same time obtain the same snapshot, and their Compute and Move are executed instantaneously. In SSYNC, we can assume that each activation defines a discrete time called *round* and Look-Compute-Move is performed instantaneously in one round. A subset of activated robots in each round is determined by an adversary and robots do not know about this subset.

As a special case of SSYNC, if all robots are activated in each round, the scheduler is called fully-synchronous (FSYNC).

In this paper, we consider ASYNC and we assume the following.

In a Look operation, a snapshot of a time t_L is taken and we say that *the Look operation is performed at time* t_L. Each Compute operation of r_i is assumed to be done at an instant time t_C and its color of light $\ell_i(t)$ and its destination des_i are assigned to the computed values at the time t_C. In a *Move* operation, when its movement begins at t_B and ends at t_E, we say that its movement is performed during $[t_B, t_E]$, its beginning and ending of the movement are denoted by $Move_{BEGIN}$ and $Move_{END}$, and its $Move_{BEGIN}$ and $Move_{END}$ occur at t_B and t_E, respectively. In the following, *Compute*, $Move_{BEGIN}$ and $Move_{END}$ are abbreviated by *Comp*, $Move_B$ and $Move_E$, respectively. When some cycle has no movement (robots change only colors of lights, or their destinations are the current positions), we can assume that the Move operation in this cycle is omitted, since we can consider the Move operation can be performed just before the next Look operation.

Also we consider the following restricted classes of ASYNC.

Let a robot execute a cycle. If any other robot cannot execute any *Look* operation between the *Look* operation and the following *Comp* operation in the cycle, its ASYNC is said to be *LC-atomic*. Thus we can assume that in LC-atomic ASYNC, *Look* and *Comp* operations in every cycle are performed at the same time. If any other robot cannot execute any *Look* operation between the $Move_B$ and the following $Move_E$, its ASYNC is said to be *Move-atomic*. In this case *Move* operations in all cycles can be considered to be performed instantaneously and at time t_M. In Move-atomic ASYNC, when a robot r observes another robot r' performing a *Move* operation at time t_M, r observes the snapshot after the moving of r'.

In our settings, robots have persistent lights and can change their colors at an instant time in each Compute operation. We consider the following robot models according to visibility of lights.

- *full-light*, the robot can recognize not only colors of lights of other robots but also its own color of light.
- *external-light*, the robot can recognize only colors of lights of other robots but cannot see its own color of light. Note that the robot can change its own color.
- *internal-light*, the robot can recognize only its own color of light but cannot see colors of lights of other robots.

Table 2. Rendezvous algorithms by robots with three kinds of lights.

Scheduler	Movement	Full-light [21]	External-light [9]	Internal-light [9]	No-light [8,17]
FSYNC	Non-rigid	−	−	−	◯
SSYNC	Non-rigid	2	3	?	×
	Rigid	−	?	6	
	Non-rigid(+δ)	−	?	3	
ASYNC	Non-rigid	3	?	?	×
	Rigid	2	12	?	
	Non-rigid(+δ)	?	3	?	

◯ and × mean solvable and unsolvable, respectively,
− indicates that this part has been solved under a weaker condition, and
? means that this part is not solved.

An n-*Gathering* problem is defined as follows. Given $n(\geq 2)$ robots initially placed at arbitrary positions in $\mathbb{R}^2$, they congregate at a single location which is not predefined in finite time. In the following, we consider the case that $n = 2$ and the 2-Gathering problem is called *Rendezvous*.

3 Previous Results for Rendezvous

Rendezvous is not solvable in the basic model without lights.

Theorem 1 (Flocchini et al. [8]). *Rendezvous is deterministically unsolvable in SSYNC even if chirality is assumed.*

If robots have a constant number of colors in their lights, Rendezvous can be solved with three kinds of lights, as shown in Table 2. Since robots are oblivious and anonymous, the colors can be used to break symmetry of robots and configurations [9,21]. Since full-lights are stronger assumptions than external-lights and internal-lights, the number of colors of full-lights used in algorithms is fewer than that of external-lights and internal-lights. Also previous algorithms and our algorithms with full-lights are in a class of algorithms called $\mathcal{L}$-algorithms [21], which means that each robot may only compute a destination point on the line connecting two points robots are located at by using only the colors of lights of current robots. The algorithms of this class are of interest because they operate also when the coordinate system of a robot is not self-consistent (i.e., it can unpredictably rotate, change its scale or undergo a reflection) [9]. On the other hand, there does not exist any $\mathcal{L}$-algorithm with external-lights and internal-lights in ASYNC and SSYNC, respectively [9].

It is still an open problem whether Rendezvous can be solved in ASYNC with 2 colors in Non-Rigid. In the following, we will show that Rendezvous is solved in ASYNC and full-light with 2 colors, if we assume (1) Rigid movement, (2) Non-Rigid movement and knowledge of the minimum distance δ robots move, (3) LC-atomic. In these cases, we can construct optimal Rendezvous algorithms with respect to the number of colors in ASYNC.

Algorithm 1. Rendezvous(scheduler, movement, initial-color)

Parameters: scheduler, movement-restriction, initial-color
Assumptions: full-light, two colors (A and B)

```
case me.light of
A:
    if other.light = A then
        me.light ← B
        me.des ← the midpoint of me.position and other.position
    else me.des ← other.position
B:
    if other.light = A then
        me.des ← me.position        // stay
    else me.light ← A
endcase
```

4 Asynchronous Rendezvous for Robots with Lights

4.1 Basic Rendezvous Algorithm

In this section, two robots are denoted by r and s. Let t_0 be the starting time of the algorithm.

Given a robot *robot*, an operation *op* ($\in \{Look, Comp, Move_B, Move_E\}$), and a time t, $t^+(robot, op)$ denotes the time *robot* performs the first *op* after t (inclusive) if there exists such an operation, and $t^-(robot, op)$ denotes the time *robot* performs the first *op* before t (inclusive) if there exists such an operation. If t is the time the algorithm terminates, $t^+(robot, op)$ is not defined for any *op*. When *robot* does not perform *op* before t and $t^-(robot, op)$ does not exist, $t^-(robot, op)$ is defined to be t_0.

A time t_c is called a *cycle start time* if the next performed operations of both r and s after t are Look operations, or otherwise, the robots performing the operations neither change their colors of lights nor move. In the latter case, we can consider that these operations can be performed before t_c, and the subsequent *Look* operation can be performed as the first operation after t_c.

Algorithm 1 is used as a basic Rendezvous algorithm which has three parameters, namely scheduler, movement restriction and an initial color of light, and it assumes full-light and uses two colors A and B.

We will show that Rendezvous(ASYNC, Rigid, A) and Rendezvous(LC-atomic ASYNC, Non-Rigid, any[3]) solve Rendezvous, and that some variant of Rendezvous(ASYNC, Non-Rigid($+\delta$), A) also solves Rendezvous.

Lemma 1. *Assume that time t_c is a cycle start time and $\ell(r, t_c) = \ell(s, t_c) = B$ in Rendezvous(ASYNC, Non-Rigid, any). If $dis(p(r, t_c)), p(s, t_c)) = 0$, then two robots r and s do not move after t_c.*

[3] Either A or B will do.

Lemma 2. *Let robot r perform a Look operation at time t in Rendezvous(ASYNC, Non-Rigid, any). If $t^-(s, Comp) \leq t$ and $\ell(r,t) \neq \ell(s,t)$, then there exists a time $t^*(> t)$ such that r and s succeed in rendezvous at time t^* by Rendezvous(ASYNC, Non-Rigid, any).*

Proof. If $\ell(r,t) = B$ and $\ell(s,t) = A$, then r does not change the color and stays at the current position. If s performs a *Look* operation at $t^+(s, Look)$, s does not change the color and s's destination is $p(r,t)$. Since both r and s do not change the colors after the time $t^+(s, Look)$, r stays at $p(r,t)$ and the destination of s is $p(r,t)$. Thus r and s succeed in rendezvous at some time $t^* \geq t^+(s, Move_E)$ even in Non-Rigid.

If $\ell(r,t) = A$ and $\ell(s,t) = B$, then r does not change the color and computes the destination as $p(s,t)$. When s finishes the *Move* operation at $t' = t^+(s, Move_E)$, s is located at $p(s,t')$. If $t' \leq t$, since r's destination is $p(s,t)$ and $p(s,t)$ is not changed (even if s performs a *Look* operation after t' and before t), r and s succeed in rendezvous at some time $t^* \geq t^+(r, Move_E)$.

Otherwise ($t < t'$), if r performs *Look* operations before t', these destinations are different since s is moving, but the color is not changed and $\ell(r,t') = A$. Since s stays at $p(s,t')$ after t', r and s succeed in rendezvous at some time $t^* \geq t^+(r, Move_E)$.

In both cases r and s do not move after t^* by the algorithm. □

4.2 ASYNC and Rigid Movement

If Rigid movement is assumed, asynchronous Rendezvous can be done with 2 colors.

Theorem 2. *Rendezvous(ASYNC, Rigid, A) solves Rendezvous.*

Proof. Let t_0 be the starting time of the algorithm and let r and s be two robots whose colors of lights are A. Without loss of generality, r is assumed to perform the *Look* operation first at time t_1, that is, $t_1 = t_0^+(r, Look)$. Let $t_2 = t_0^+(s, Look)$. There are two cases: (I) $t_1 \leq t_2 < t_0^+(r, Comp)$ and (II) $t_0^+(r, Comp) \leq t_2$.

Case (I): Since $\ell(s,t_2) = A$ and $\ell(r,t_2) = A$, s moves to the midpoint of $p(r,t_0)$ and $p(s,t_0)$ at time $t_3 = t_0^+(s, Move_E)$. Robot s changes its color of light from A to B at time $t_0^+(s, Comp)$ and $\ell(s,t_3) = B$. There are two cases (I-1) $t_3' = t_0^+(r, Move_E) < t_3$ and (I-2) $t_3 \leq t_3' = t_0^+(r, Move_E)$.

Case (I-1): Robot r reaches the destination at time t_3' but s does not reach the destination ($t_3' = t_0^+(r, Move_E) < t_3$). If r does not perform any operations during $[t_3', t_3]$, t_3 becomes a cycle start time and then r and s rendezvous at time t_3 and the two robots do not move after t_3 by Lemma 1.

Otherwise, r performs several operations during $[t_3', t_3]$. Suppose r performs at least one *Look* operation and one *Comp* operation during $[t_3', t_3]$ and t_C denotes the time r performs the *Comp* operation ($t_C =$

$t_3'^{+}(r, Comp)$). Note that r only changes its color of light and does not move in this cycle. Then its color of light is changed to A at t_C, and $\ell(r, t_C) = A$ and $\ell(s, t_C) = B$. Thus, the next *Look* operation of r or s after t_C satisfies the conditions of Lemma 2, and r and s succeed in rendezvous. The remaining case is that r performs only a *Look* operation during $[t_3', t_3]$. Let t_L be the time r performs the *Look* operation. Since r observes $\ell(s, t_L) = B$ and r and s are located at the same point at t_3, this case is the same as the first case.

Case (I-2): Interchanging the roles of r and s, this case can be reduced to Case (I-1).

Case (II): Since $(t_0)^{+}(r, Comp) \leq t_2$ and $\ell(r, t_2) \neq \ell(s, t_2)$, r and s succeed in rendezvous by Lemma 2. □

Note that this algorithm does not terminate and we cannot change the algorithm so that the fixed one can terminate. It is an open problem whether there exists an algorithm which solves Rendezvous and terminates with two colors in ASYNC. Also there exists an execution such that Rendezvous(Async, Rigid, any) does not work in general. In fact, if the initial colors of lights for both robots are B, this algorithm cannot solve Rendezvous. Figure 1 shows a counterexample where Rendezvous(Async, Rigid, B) does not work. Since the colors of lights at t_5 are B, this execution repeats forever and achieves only convergence, that is, the robots move arbitrarily close to each other, but might not rendezvous within finite time. This counterexample also shows that Rendezvous (Move-atomic ASYNC, Rigid, B) does not work. However, if we assume LC-atomic ASYNC, we can show that Rendezvous(LC-atomic ASYNC, Rigid, B) solves Rendezvous.

Lemma 3. *Rendezvous(LC-atomic ASYNC, Rigid, B) solves Rendezvous.*

Proof. In LC-atomic ASYNC, any *Look* operation and the following *Comp* operation are performed at the same time and this operation is denoted by LC. Let t_c be a cycle start time and let r perform an LC operation first and let $t_1 = t_c^{+}(r, LC)$. There are two cases: (I) $t_1 = t_c^{+}(s, LC)$ and (II) $t_1 < t_c^{+}(s, LC) = t_2$.

Case (I): In this case, since t_1 becomes a cycle start time and $\ell(r, t_1) = \ell(s, t_1) = A$, the lemma holds by Theorem 2.

Case (II): In this case, since $\ell(r, t_1) = A$ and $\ell(s, t_2) = B$, the lemma holds by Lemma 2. □

In an execution of Rendezvous(Async, Non-Rigid, A) starting from $\ell(r, t_0) = \ell(s, t_0) = A$, if r and s perform one cycle simultaneously, their colors can be changed into B without attaining Rendezvous, that is, there is a cycle start time $t_c (\geq t_0)$ such that $\ell(r, t_c) = \ell(s, t_c) = B$. Thus it cannot solve Rendezvous even if both initial colors of lights are A. In Subsect. 4.3, we will show that if ASYNC is restricted to LC-atomic, Rendezvous can be solved in Non-Rigid with two colors from any initial colors of lights.

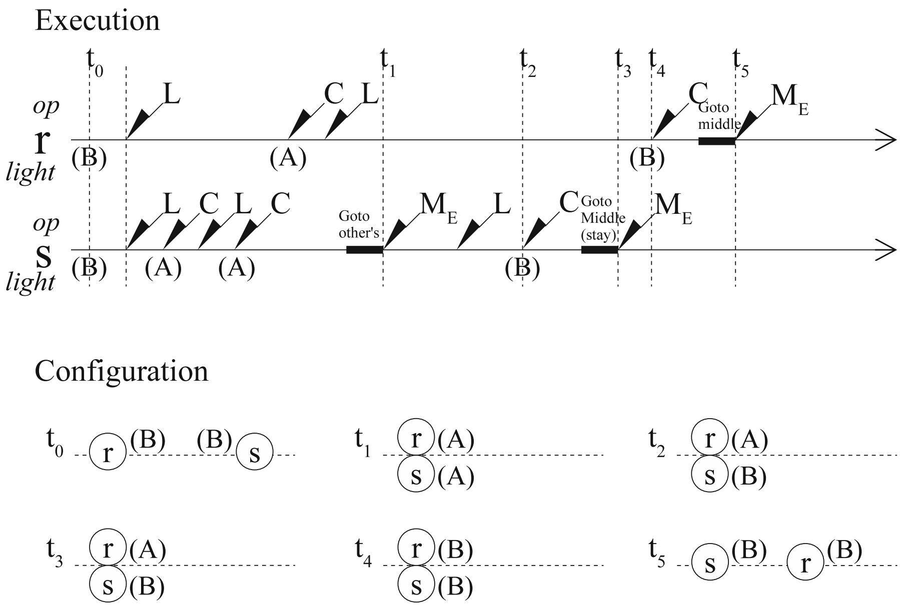

Fig. 1. Rendezvous(ASYNC, Rigid, B) cannot solve Rendezvous in general.

4.3 LC-Atomic ASYNC and Non-Rigid Movement

Let t_c be a cycle start time of the algorithm. There are three cases according to the colors of lights of two robots r and s, (I) $\ell(r, t_c) \neq \ell(s, t_c)$, (II) $\ell(r, t_c) = \ell(s, t_c) = A$, and (III) $\ell(r, t_c) = \ell(s, t_c) = B$.

Lemma 4. *If $\ell(r, t_c) \neq \ell(s, t_c)$ and the algorithm starts at t_c, then there exists a time $t^*(\geq t)$ such that r and s succeed in rendezvous at time t^* by Rendezvous (LC-atomic ASYNC, Non-Rigid, any).*

Proof. This is obvious from Lemma 2. □

Lemma 5. *If $\ell(r, t_c) = \ell(s, t_c) = B$ and the algorithm starts at t_c, then there exists a time $t^*(\geq t)$ such that r and s succeed in rendezvous at time t^* by Rendezvous(LC-atomic ASYNC, Non-Rigid, any) or t^* is a cycle start time and $\ell(r, t^*) = \ell(s, t^*) = A$.*

Proof. Let r perform the LC operation first and let $t_1 = t_c^+(r, LC)$. There are two cases: (I) $t_1 = t_c^+(s, LC)$ and (II) $t_1 < t_c^+(s, LC) = t_2$.

(I) In this case, $\ell(r, t_1) = \ell(s, t_1) = B$ and there are two cases: (I-1) $t_1^+(r, LC) \neq t_1^+(s, LC)$ and (I-2) $t_1^+(r, LC) = t_1^+(s, LC)$.

(I-1) This case can be proven by Lemma 2.

(I-2) Letting $t^* = t_1^+(r, LC) = t_1^+(s, LC)$, $\ell(r, t^*) = \ell(s, t^*) = A$ and t_* becomes a cycle start time. Also at the time t_* rendezvous is succeeded or $dis(p(r, t^*), p(s, t^*)) \leq dis(p(r, t_c), p(s, t_c)) - 2\delta$.

(II) Since $\ell(r, t_1) = A$ and $\ell(s, t_2) = B$, this case is proven by Lemma 2. □

Algorithm 2. RendezvousWithDelta (ASYNC, Non-Rigid ($+\delta$), A)

```
Assumptions: full-light, two colors (A and B)
case dis(me.position, other.position)(= DIST) of
DIST > 2δ:
   if me.light = other.light = B then
      me.des ← the point moving by δ/2 from me.position to other.position
   else me.light ← B
2δ ≥ DIST ≥ δ:
   if me.light = other.light = A then
      me.light ← B
      me.des ← the midpoint of me.position and other.position
   else me.light ← A
δ > DIST:      // Rendezvous(ASYNC, Rigid, A)
   case me.light of
   A:
      if other.light = A then
         me.light ← B
         me.des ← the midpoint of me.position and other.position
      else me.des ← other.position
   B:
      if other.light = A then me.des ← me.position      // stay
      else me.light ← A
   endcase
endcase
```

Lemma 6. *If $\ell(r, t_c) = \ell(s, t_c) = A$ and the algorithm starts at t_c, then there exists a time $t^*(\geq t_c)$ such that r and s succeed in rendezvous at time t^* by Rendezvous(LC-atomic ASYNC, Non-Rigid, any) or t^* is a cycle start time, $\ell(r, t^*) = \ell(s, t^*) = A$ and $dis(p(r, t^*), p(s, t^*)) \leq dis(p(r, t_c), p(s, t_c)) - 2\delta$.*

Proof. Let $t_1 = t_c^+(r, LC)$ and $t_c^+(s, LC)$. If $t_1 = t_2$, then letting $t^* = t_1$, t^* is a cycle start time and $\ell(r, t^*) = \ell(s, t^*) = A$. Otherwise, Lemma 2 proves this case. □

Lemmata 4 to 6 imply the next theorem.

Theorem 3. *Rendezvous(LC-atomic ASYNC, Non-Rigid, any) solves Rendezvous.*

4.4 ASYNC and Non-Rigid Movement($+\delta$)

Although it is still open whether asynchronous Rendezvous can be solved in Non-rigid with two colors of lights, if we assume Non-Rigid($+\delta$), we can solve Rendezvous modifying Rendezvous(ASYNC, Non-Rigid($+\delta$), A) and using the minimum moving value δ in it.

Let $dist_0 = dis(p(r, t_0), p(s, t_0))$ and let RendezvousWithDelta (Algorithm 2) begin with $\ell(r, t_0) = \ell(s, t_0) = A$. If $dist_0 > 2\delta$, both robots do not move until both colors of lights become B (**lines** 3-5) and there exists a cycle start time $t_1(> t_0)$ such that $\ell(r, t_1) = \ell(s, t_1) = B$. After $\ell(r, t_1) = \ell(s, t_1) = B$, the distance between r and s is reduced by $\delta/2$ without changing the colors of lights (**line** 4) and the distance falls in $[2\delta, \delta]$ and both colors of lights become A at a cycle starting time t_2. After $\ell(r, t_2) = \ell(s, t_2) = A$, we can use Rendezvous(ASYNC, Rigid, A) since $2\delta \geq dis(p(r, t_2), p(s, t_2)) \geq \delta$. Therefore, rendezvous is succeeded. Note that in Algorithm 2, the initial pair of colors of r and s is $(\ell(r, t_0), \ell(s, t_0)) = (A, A)$ and it is changed into $(\ell(r, t_1), \ell(s, t_1)) = (B, B)$ without changing the distance of r and s. And it is changed into $(\ell(r, t_2), \ell(s, t_2)) = (A, A)$ when the distance becomes between δ and 2δ. These *mode* changes are necessary and our algorithm does not work correctly if these mode changes are not incorporated in the algorithm. If the algorithm starts with $(\ell(r, t_0), \ell(s, t_0)) = (A, A)$ and does not use the mode change, when the distance becomes between δ and 2δ at some time t', the configuration becomes $(\ell(r, t'), \ell(s, t')) = (B, B)$ and therefore Rendezvous fails from the configuration.

Lemma 7. *If $dist_0 > 2\delta$, in any execution of RedezvousWithDelta(ASYNC, Non-Rigid(+δ), A),*

(1) there exists a cycle start time $t_1(> t_0)$ such that $\ell(r, t_1) = \ell(s, t_1) = B$ and $dis(p(r, t_1), p(s, t_1)) = dist_0$, and
(2) there exists a cycle start time $t_2(> t_1)$ such that $\ell(r, t_2) = \ell(s, t_2) = A$ and $2\delta \geq dis(p(r, t_2), p(s, t_2)) \geq \delta$.

Proof. (1) Without loss of generality, r performs the *Look* operation first and let t_{rL} be such a time. The color of r is changed from A to B at a time t_{rC}. Since $\ell(s, t_0) = A$, s performs a *Look* operation at a time $t_{sL}(\geq t_{rL})$ and changes its color from A to B at a time t_{sC}. Then, let $t_1 = \max(t_{rC}, t_{sC})$. If a *Comp* operation is performed immediately after t_1, the robot does not change its color of light, since the robot performs the preceding *Look* operation before t_1. Thus, t_1 becomes a cycle start time.

(2) Since t_1 is a cycle start time, we can assume that the algorithm starts at t_1 with $\ell(r, t_1) = \ell(s, t_1) = B$. The distance $dist_0$ is reduced by $\delta/2$ every cycle of each robot after t_1. Since $dist_0 > 2\delta$, $dist_0$ can be denoted by $x(\delta/2) + \varepsilon$, where $x \geq 4$ and $0 \leq \varepsilon < \delta/2$.

Let t be the time of the $(x-2)$-th *Look* operation among *Look* operations r and s performed after t_1, and without loss of generality, let r be the robot performing the $(x-2)$-th *Look* operation. Note that among $(x-3)$ *Move* operations between t_1 and t at least $\max(0, x-4)$ *Move* operations have been completed and at most one *Move* operation has not been completed yet.

Let $t' = t^-(s, Look)$.[4] We have two situations (Fig. 2). One is that (I) $(x-3)$ *Move* operations are completed until t. This case satisfies $2\delta \geq dis(p(r, t), p(s, t)) \geq \delta$. The other is that (II) the $(x-3)$-th *Move* operation s performs has not been completed at t.[5] The latter case is divided into two cases

[4] If s performed no *Look* operations after t_1, $t' = t_1$.
[5] This case includes $t < t'^+(s, Move_B)$.

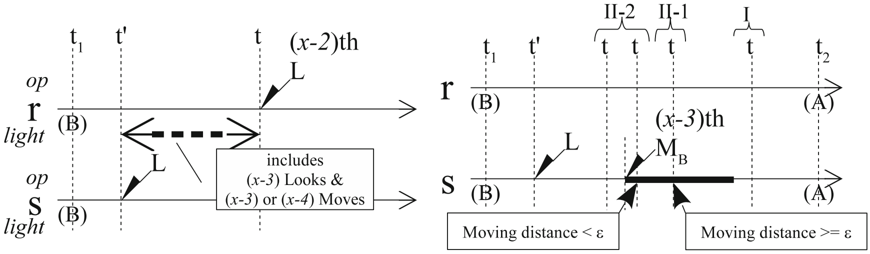

Fig. 2. Situations in the proof of Lemma 7

(II-1) $2\delta \geq dis(p(r,t), p(s,t)) \geq \delta$ and (II-2) $dis(p(r,t), p(s)) > 2\delta$, according to the time r performs the *Look* operation.

Case (I) and (II-1): Since $2\delta \geq dis(p(r,t), p(s,t)) \geq \delta$, r changes its color of light to A at $t^+(r, Comp)$. When s performs a *Look* operation at $t_{sL} = t^+(s, Look)$, s observes $2\delta \geq dis(p(r, t_{sL}), p(s, t_{sL})) \geq \delta$ and $\ell(s, t_{sL}) = B$, and changes its color of light to A at $t_{sC} = t^+(s, Comp)$. Letting $t_2 = \max(t_{rC}, t_{sC})$, t_2 becomes a cycle start time as follows.
When $t_2 = T_{rC}$, s changes its color of light to A at $t_{sC}(\leq t_{rC})$. Even if s performs a *Look* operation at t_L after t_{sC} before $t_2 = t_{rC}$, s does not change its color at $t_L^+(s, Comp)$ since $\ell(r, t_L) = B$. The case that $_2 = T_{sC}$ can be proven similarly.

Case (II-2): Since $dis(p(r,t), p(s,t)) > 2\delta$, r reduces the distance by $\delta/2$. Then, r performs the *Move* operation and subsequently performs the next *Look* operation at $t_{rL} = t^+(r, Look)$. After that it changes its color of light to A at $t_{rC} = t^+(r, Comp)$, since $\delta \leq dis(p(r, t_{rL}), p(s, t_{rL})) \leq 2\delta$. The next *Look* operation of s is performed after $t'^+(s, Move_E)$, and s changes its color of light to A at $t_{sC} = t^+(s, Comp)$. Robot s changes its color of light to A at t_{sC}. Letting $t_2 = \max(t_{rC}, t_{sC})$, we can prove that t_2 becomes a cycle start time similar to the former case. □

The following two lemmata can be proven similarly to the proof of Theorem 2.

Lemma 8. *If $2\delta \geq dist_0 \geq \delta$, then RedezvousWithDelta(ASYNC, Non-Rigid (+δ), A) solves Rendezvous.*

Lemma 9. *If $dist_0 > \delta$, then RedezvousWithDelta(ASYNC, Non-Rigid (+δ), A) solves Rendezvous.*

Lemmata 7 to 9 imply the following theorem.

Theorem 4. *RedezvousWithDelta(ASYNC, Non-Rigid(+δ), A) solves Rendezvous.*

5 Concluding Remarks

We have shown that Rendezvous can be solved in ASYNC with the optimal number of colors of lights if Non-Rigid($+\delta$) movement is assumed. We have also shown that Rendezvous can be solved by an $\mathcal{L}$-algorithm in ASYNC and Non-Rigid with the optimal number of colors of lights if ASYNC is LC-atomic. Interesting open problems are whether can Rendezvous be solved in ASYNC and Non-Rigid with 2 colors or not,[6] and what condition of ASYNC can $\mathcal{L}$-algorithms be solved in Non-Rigid with 2 colors?

Acknowledgment. This work is supported in part by KAKENHI no. 17K00019 and 15K00011.

References

1. Agmon, N., Peleg, D.: Fault-tolerant gathering algorithms for autonomous mobile robots. SIAM J. Comput. **36**, 56–82 (2006)
2. Bouzid, Z., Das, S., Tixeuil, S.: Gathering of mobile robots tolerating multiple crash faults. In: Proceedings of 33rd ICDCS (2013)
3. Cieliebak, M., Flocchini, P., Prencipe, G., Santoro, N.: Distributed computing by mobile robots: gathering. SIAM J. Comput. **41**(4), 829–879 (2012)
4. Das, S., Flocchini, P., Prencipe, G., Santoro, N., Yamashita, M.: Autonomous mobile robots with lights. Theor. Comput. Sci. **609**, 171–184 (2016)
5. Défago, X., Potop-Butucaru, M.G., Clément, J., Messika, S., Raipin Parvédy, P.: Fault and byzantine tolerant self-stabilizing mobile robots gathering - feasibility study - CoRRabs/1602.05546 (2016)
6. Degener, B., Kempkes, B., Langner, T., Meyer auf der Heide, F., Pietrzyk, P., Wattenhofer, R.: A tight runtime bound for synchronous gathering of autonomous robots with limited visibility. In: Proceedings of 23rd ACM SPAA, pp. 139–148 (2011)
7. Dieudonné, Y., Petite, F.: Self-stabilizing gathering with strong multiplicity detection. Theor. Comput. Sci. **428**(13), 47–57 (2012)
8. Flocchini, P., Prencipe, G., Santoro, N.: Distributed Computing by Oblivious Mobile Robots, Synthesis Lectures on Distributed Computing Theory. Morgan & Claypool, San Rafael (2012)
9. Flocchini, P., Santoro, N., Viglietta, G., Yamashita, M.: Rendezvous with constant memory. Theor. Comput. Sci. **621**, 57–72 (2016)
10. Flocchini, P., Prencipe, G., Santoro, N., Widmayer, P.: Gathering of asynchronous robots with limited visibility. Theor. Comput. Sci. **337**(1–3), 147–168 (2005)
11. Heriban, A., Défago, X., Tixeuil, S.: Optimally gathering two robots. Research Report HAL Id: hal-01575451. UPMC Sorbonne Universités, August 2017
12. Izumi, T., Bouzid, Z., Tixeuil, S., Wada, K.: Brief announcement: the BG-simulation for byzantine mobile robots. In: Peleg, D. (ed.) DISC 2011. LNCS, vol. 6950, pp. 330–331. Springer, Heidelberg (2011). https://doi.org/10.1007/978-3-642-24100-0_32

[6] Very recently it has been solved affirmatively [11].

13. Izumi, T., Katayama, Y., Inuzuka, N., Wada, K.: Gathering autonomous mobile robots with dynamic compasses: an optimal result. In: Pelc, A. (ed.) DISC 2007. LNCS, vol. 4731, pp. 298–312. Springer, Heidelberg (2007). https://doi.org/10.1007/978-3-540-75142-7_24
14. Izumi, T., et al.: The gathering problem for two oblivious robots with unreliable compasses. SIAM J. Comput. **41**(1), 26–46 (2012)
15. Kamei, S., Lamani, A., Ooshita, F., Tixeuil, S.: Asynchronous mobile robot gathering from symmetric configurations without global multiplicity detection. In: Kosowski, A., Yamashita, M. (eds.) SIROCCO 2011. LNCS, vol. 6796, pp. 150–161. Springer, Heidelberg (2011). https://doi.org/10.1007/978-3-642-22212-2_14
16. Lin, J., Morse, A.S., Anderson, B.D.O.: The multi-agent rendezvous problem. Parts 1 and 2. SIAM J. Control Optim. **46**(6), 2096–2147 (2007)
17. Prencipe, G.: Impossibility of gathering by a set of autonomous mobile robots. Theor. Comput. Sci. **384**(2–3), 222–231 (2007)
18. Souissi, S., Défago, X., Yamashita, M.: Using eventually consistent compasses to gather memory-less mobile robots with limited visibility. ACM Trans. Auton. Adapt. Syst. **4**(1), 1–27 (2009)
19. Suzuki, I., Yamashita, M.: Distributed anonymous mobile robots: formation of geometric patterns. SIAM J. Comput. **28**, 1347–1363 (1999)
20. Terai, S., Wada, K., Katayama, Y.: Gathering problems for autonomous mobile robots with lights. Technical report of Wada Labo., Hosei University, TRW-16-1 (2016)
21. Viglietta, G.: Rendezvous of two robots with visible bits. Technical report arXiv:1211.6039 (2012)