法政大学学術機関リポジトリ
HOSEI UNIVERSITY REPOSITORY

PDF issue: 2025-07-10

状態を持つ2台の自律分散ロボットランデブーアルゴリズムに関する研究

OKUMURA, Takashi / 奥村, 太加志

(出版者 / Publisher)
法政大学大学院理工学研究科
(雑誌名 / Journal or Publication Title)
法政大学大学院紀要. 理工学・工学研究科編
(巻 / Volume)
59
(開始ページ / Start Page)
1
(終了ページ / End Page)
8
(発行年 / Year)
2018-03-31
(URL)
https://doi.org/10.15002/00021583

状態を持つ 2 台の自律分散ロボットのランデブーアルゴリズムに関する研究

RENDEZVOUS OF AUTONOMOUS MOBILE ROBOTS WITH STATES

奥村太加志
Takashi OKUMURA
指導教員　和田幸一

法政大学大学院理工学研究科応用情報工学専攻修士課程

We study a *Rendezvous* problem for 2 autonomous mobile robots in asynchronous setting with persistent memory called *light*. It is known that Rendezvous is impossible when robots have no lights in basic common models, even if the system is semi-synchronous. We show that Rendezvous can be solved with optimal number of states if we consider some restricted class of asynchronous setting. In full-light (the robot can recognize own states and states of others), Rendezvous can be solved with 2 states. In external-light, (the robot can recognize only states of others) Rendezvous can be solved with 4 states or 5 states in self-stabilization. When giving restrictions on the initial states and movement of the robots, Rendezvous can be solved with 3 states in external-light.

***Key Words** : Autonomous Mobile Robots, Rendezvous, Asynchronous*

１．はじめに

複数の計算機が自律的に動作し，互いに通信し合うことで全体である目的を達成するシステムのことを分散システムという．その分野の一つで，複数のロボットが自律的に計算，移動を行うことで，全体で一つの問題を解決するシステムのことを自律分散ロボット群という．自律分散ロボット群の研究では，与えられた問題を解くのに必要な能力を明らかにするため，理論モデルを用いて問題の可解性を調べるのが主流である[1-3,6,11, 17,21].

理論モデルにおいて，ロボットは平面上を自由に移動できる匿名な点とみなし，それぞれ局所座標を持っている．局所座標について，それぞれの座標軸の向きの一致や時計回りの方向に関する合意(キラリティ)があると能力が向上することが知られているが[14,15,18]，本稿では考えないものとする．

ロボットは同一のアルゴリズムを実行し，任意の時刻に動作と待機を行う．動作すると，周囲のロボットの観測(Look 命令)，観測結果に基づいて移動先を計算(Compute 命令),計算された目的地へ移動(Move 命令)の 3 つの命令を 1 つのサイクルとして実行する．Look 命令に関して，視界範囲に制限のあるモデルが考案されているが[6,10,16]，本稿のロボットに視界の制限はないものとする．Move 命令に関して，一度の命令で計算された目的地へ必ずたどり着くものを Rigid といい，一度の命令でたどり着かない場合があり，その時は少なくとも最小移動距離$\delta(>0)$は必ず移動するものとする．この移動を Non-rigid という．Non-rigid のうち，ロボットが最小移動距離δの値に関する知識を持っているものを Non-rigid$(+\delta)$という．スナップショットはサイクル実行毎に削除され，過去の履歴は持たないものとする．これを無記憶という．

ロボットは 3 つの命令を実行する時刻を決定するスケジューラに従って動作する．全てのロボットの命令を行う時刻が共通であるものを全同期(FSYNC)，FSYNC と同様だが，サイクルを実行しないロボットの存在を 1 台以上許すものを半同期(SSYNC)，全てのロボットが独立して動作し，同期の仮定を考えないものを非同期(ASYNC)という．本稿では ASYNC に制限を与えたクラスを 2 つ提案する．1 つ目はロボットが実行した際，Look 命令と Compute 命令を同時刻に実行できるものとする(この時の命令を LC 命令と呼ぶ)．このクラスを LC-atomic ASYNC という．2 つ目は，Move 命令が瞬間的に実行されるものとする．このクラスを Move-atomic ASYNC と呼ぶ.

その他の仮定として,同一の点に存在するロボットが 1 台か複数かを認識できる多重検知があるが[2,7,8]，本稿では扱わない.

ロボットに与える問題として，一点集合問題がある[2,3,5,6,11,13-15,18]．これは複数のロボットが任意の

初期位置から有限時間内に予め決められていない一点に集合するという問題である．ロボットの台数が 2 台の場合，特別にランデブー問題という．一点集合問題は FSYNC のロボットでは可解であるが，SSYNC や ASYNC のロボットでは非可解であることが知られている．このように基本的な理論モデルでは非可解となる問題が知られている．そこで基本的な機能に加えて，自身の状態を記録できる定数ビットの記憶領域(*light*)を搭載したモデルが考案されている[4]．light で示した状態は Look 命令で観測し，Compute 命令で更新することができる．観測時に，自身の状態のみ観測できるものを internal-light，他のロボットの状態のみを観測できるものを external-light，双方の状態を観測できるものを full-light という[9]．目的地を計算する際，互いの状態に関する情報のみを用いて決定するアルゴリズムのクラスを$\mathcal{L}$という[9,20]．また，ロボットの初期状態は少なくとも 1 つの同じ状態から開始するものとする．この時，任意の同じ初期状態から開始して正しく動作するアルゴリズムを準自己安定，任意の状態から開始しても正しく動作するアルゴリズムを自己安定という．

従来の結果より，状態付き SSYNC，ASYNC でランデブー問題が可解となる場合がある(表 1)．

本稿では，LC-atomic ASYNC について考え，ランデブー問題を最適な状態数で解くアルゴリズムを提案する．full-light の場合，Non-rigid，状態数 2 でランデブー問題を解く，クラス$\mathcal{L}$に属する自己安定なアルゴリズムを提案する．external-light の場合，Rigid，状態数 3 でランデブー問題を解くアルゴリズムを提案する．この時，状態数 4 ならば Non-rigid でも正しく動作する準自己安定なアルゴリズム，状態数 5 ならば任意の初期状態でも正しく動作する自己安定なアルゴリズムを提案する．[9]において，SSYNC，external-light，Non-rigid，状態数 3 でランデブー問題を解くアルゴリズムが知られているが，この状態数が最適であることを証明する．

表 1. 状態付きロボットによるランデブーアルゴリズム

		full	external	internal	−
FSYNC	NR				○
SSYNC	(+δ)			3	×
	R			6	
	NR	2	3	∞	
ASYNC	(+δ)		3	?	×
	R	2	12	?	
	NR	3	∞	?	

NR ： Non-rigid，R ： Rigid，(+δ) ： Non-rigid(+δ)とする．状態数に関わらず非可解である場合は∞とする．より弱い仮定ですでに可解である場合は斜線とする．可解となるアルゴリズムが知られていない場合は?とする．

２．モデルと定義

（１）ロボットのモデル

システムは n 台の匿名なロボットの集合$\mathcal{R} = \{r_1, \ldots, r_n\}$で構成され，2 次元平面$\mathbb{R}^2$に存在する点として扱う．各ロボットは$\mathbb{R}^2$内を共通のアルゴリズムに従って自律的に動作することができる．

各ロボットには自身のいる位置が常に原点となる独自の座標系を持っている．この座標系は他のロボットと合意する必要はない．つまりロボット同士に単位距離や座標系の向きなどの共通の知識，時計回りの方向に関する合意(キラリティ)はない．

任意の時刻において，ロボットは動作と待機を行うことができる．ロボットが動作すると，以下の 3 つの命令をサイクルとして実行し，実行後は待機，または新たなサイクルを実行する．このサイクルを Look-Compute-Move サイクルという．

- Look 命令 ： ロボットは自身のセンサを使用し，自身の座標系に従ってロボットの位置を得る．この時，ロボットは$\mathbb{R}^2$内全てのロボットを観測できるものとする．得た情報はスナップショットとして保存される．この命令は瞬間的に行われる．
- Compute 命令 ： スナップショットを入力としてアルゴリズムを実行し，目的地を計算する．スナップショットは使用後に削除される(無記憶)．この命令は瞬間的に行われる．
- Move 命令 ： 計算された目的地に移動する．一度の命令で目的地にたどり着くものを Rigid という．反対に，一度の目的地で目的地にたどり着かない場合があり，その時は少なくとも最小移動距離δ(> 0)は必ず移動するものとする．この移動を Non-rigid という．また Non-rigid だが，ロボットが最小移動距離δの値に関する知識を持っているものを Non-rigid(+δ)という．各ロボットの移動速度に一貫性はないものとし，命令には速度に応じた時間がかかるものとする．

ロボットはスケジューラに従って，各命令を実行する時刻が決定される．各命令の同期の程度によって，以下の 3 つのスケジューラが定義できる．

- ASYNC (非同期) ： 各ロボットは独立して動作している．
- SSYNC (半同期) : システム内の一部，または全てのロボットがサイクル内の各命令を同時刻に実行する．サイクルを実行しないロボットは待機とする．
- FSYNC (全同期) ： システム内の全てのロボットがサイクル内の各命令を同時刻に実行する．

これら 3 つのスケジューラについて，無期限に待機となるロボットは存在しない．このようなスケジューラを

公平なスケジューラであるという．

本稿ではこの基本的な理論モデルに加えて，定数ビットの記憶領域(*light*)を持つモデルを扱う[4]. light は複数の色を示すことができ，これによって状態を表現する．状態は Look 命令時に観測，Compute 命令時に更新することができる．ロボットの状態に関する可視性について、以下の 3 つのモデルが定義できる．

- full-light ： 自身を含めた全てのロボットの状態を観測できる．
- external-light [9]： 自身以外全てのロボットを観測できる．
- internal-light [9]： 自身の状態のみ観測できる．

本稿では ASYNC について，新たに制限を与えたクラスを 2 つ提案する. 1 つ目はロボットがサイクルを実行した際, Look 命令と Compute 命令間に他のロボットは Look 命令を実行できないものとする．このような ASYNC を LC-atomic ASYNC という. 従って LC-atomic ASYNC では，各サイクルにおいて Look 命令と Compute 命令が同時刻に行われるものと考えることができる．この時の命令を LC 命令と呼ぶ. つまりロボットは LC 命令によって状態遷移を行ったロボットか，LC 命令を実行する前のロボットのみ観測できる(つまりスナップショットを取得したが，それに応じた状態には遷移していない曖昧なロボットは存在しない)．2 つ目はロボットがサイクルを実行した際，Move 命令は瞬間的に行われるものとする．このような ASYNC を Move-atomic ASYNC という. 従って Look 命令を実行したロボットは Move 命令によって移動を完了したロボットか，Move 命令を実行する前のロボットのみ観測できる(つまり移動中のロボットを観測することはない).

（２）アルゴリズムに関する定義

アルゴリズムによってロボットが目的地を計算する際，周囲の他のロボット位置を基準とすることが考えられる．ロボットが 2 台の場合，自身と相手のロボットの位置の直線上を移動し，その目的地をロボットの状態によって決定することが考えられる．よって以下の形式で示すことができる．

$$(1-\lambda)\cdot me.light+\lambda\cdot other.light \qquad (1)$$

$\lambda\in\mathbb{R}$とし, λの値は自身の示す状態(*me.light*)と相手の示す状態(*other.light*)のみで決定する．状態遷移についても，自身と相手の状態のみで決定する．このようなアルゴリズムのクラスを$\mathcal{L}$と定義する[9,20].

ロボットの初期状態について，正しいアルゴリズムは少なくとも 1 つの同じ状態から開始して問題を解いている．つまり，初期状態が異なる場合のみ正しく動作するアルゴリズムは考えないものとする．この時，任意の同じ初期状態から開始して正しく動作するものを準自己安定なアルゴリズムという．また，任意の初期状態から開始して正しく動作するものを自己安定なアルゴリズムという．

（３）ランデブー問題

n(≥ 2)台のロボットが$\mathbb{R}^2$内の任意の初期位置から，有限時間内に予め決められていない一点に集合する問題のことを一点集合問題という．n = 2の場合の一点集合問題を特別にランデブー問題という．本稿ではランデブー問題について扱う．

３． 従来の結果

FSYNC ではランデブー問題を解くアルゴリズムは存在するが，SSYNC で解くアルゴリズムは存在しないことが知られている．

定理１ [10] キラリティを仮定したとしても, SSYNC でランデブー問題を解くアルゴリズムは存在しない.

ロボットが状態を持つ場合，以下の条件よりランデブー問題を解くアルゴリズムが知られている．

定理２ [9,12,20] ロボットが状態を持つ場合，以下の条件よりランデブー問題を解くアルゴリズムが知られている．

- SSYNC, internal-light, Rigid の場合，状態数 6 でランデブー問題を解くアルゴリズム
- SSYNC, internal-light, Non-rigid(+δ)の場合, 状態数 3 でランデブー問題を解くアルゴリズム
- ASYNC, external-light, Rigid の場合，状態数 12 でランデブー問題を解くアルゴリズム
- ASYNC, external-light, Non-rigid(+δ)の場合，状態数 3 でランデブー問題を解くアルゴリズム

ランデブー問題を解くアルゴリズムのうち，以下の条件ではクラス$\mathcal{L}$に属するものが存在する．

- SSYNC, full-light, Non-rigid の場合，状態数 2 でランデブー問題を解く自己安定なアルゴリズム
- SSYNC, external-light, Non-rigid の場合，状態数 3 でランデブー問題を解く自己安定なアルゴリズム
- ASYNC, full-light, Non-rigid の場合，状態数 3 でランデブー問題を解く自己安定なアルゴリズム
- ASYNC, full-light, Rigid の場合, 状態数 2 でランデブー問題を解く自己安定なアルゴリズム

定理３ [9,20] 以下の条件では，ランデブー問題を解くアルゴリズムが存在しないことが知られている．

- ASYNC, full-light, Rigid, 状態数 2 でランデブー問題を解く，クラス$\mathcal{L}$に属する自己安定なアルゴリズム
- ASYNC, full-light, Non-rigid, 状態数 2 でランデブー問題を解く，クラス$\mathcal{L}$に属するアルゴリズム
- SSYNC, internal-light, 状態数無制限でランデブー

問題を解く，クラス$\mathcal{L}$に属するアルゴリズム

- ASYNC, external-light, 状態数無制限でランデブー問題を解く，クラス$\mathcal{L}$に属するアルゴリズム

４．full-light によるランデブー問題

（１）LC-atomic ASYNC, Non-rigid の場合

定理２より，ASYNC, full-light, Rigid, 状態数 2 でランデブー問題を解く自己安定なアルゴリズムが知られている[20]．そのアルゴリズムを以下に示す．

Algorithm 1 Rendezvous(scheduler, movement, initial-light)

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

```
1:  case me.light of
2:  A:
3:      if other.light = A then
4:          me.light ← B
5:          me.des ← the midpoint of me.position and other.position
6:      else me.des ← other.position
7:  B:
8:      if other.light = A then
9:          me.des ← me.position //stay
10:     else me.light ← A
11: end case
```

Algorithm1 は，スケジューラ，移動性，初期状態の 3 つのパラメータを持っている．自己安定だと仮定しても正しく動作する場合は初期状態に *SS* と表記し，準自己安定だと仮定しても正しく動作する場合は *qS* と表記する．自身と相手が状態 A である場合，状態 B となり 2 台の中点に移動する．自身の状態が A，相手の状態が B である場合，相手の位置へ移動する．自身と相手が状態 B である場合，状態を A に変える．Algorithm1 の状態遷移図を図 1 に示す．

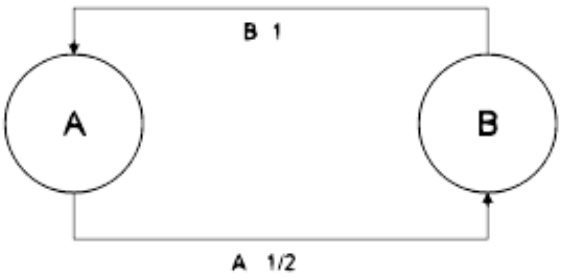

図 1．Algorithm1 の状態遷移図
ノードのラベルは自身の状態を示している．矢印のラベルは相手の状態と，それを観測したときの移動距離を示している(1/2 : 2 台の中点，1：相手の位置，0：停止)．

[20]より以下の定理が知られている．

定理４　[20] Algorithm1 について，Rendezvous(ASYNC, Rigid, A)はランデブー問題を正しく解く．

定理５　[20] Algorithm1 について，Rendezvous(SSYNC, Non-rigid, *SS*)はランデブー問題を正しく解く

Algorithm1 について，Rendezvous(Move-atomic ASYNC, Rigid, B)の場合，正しく動作しない．そのため，より仮定の弱い Rendezvous(ASYNC, Rigid, *SS*)も正しく動作しない．しかし，Rendezvous(LC-atomic ASYNC, Non-rigid, B)については正しく動作する．同様に Rendezvous(LC-atomic ASYNC, Non-rigid, A)も正しく動作することを証明した．よって以下の定理が成り立つ．

定理６　Algorithm1 について，Rendezvous(LC-atomic ASYNC, Non-rigid, *SS*)はランデブー問題を正しく解く．

（２）ASYNC, Non-rigid(+δ)の場合

ASYNC, full-light, Non-rigid, 状態数 2 でランデブー問題を正しく解くアルゴリズムは存在しない[1]．本稿では，Non-rigid(+δ)とした場合，状態数 2 でランデブー問題を正しく解くアルゴリズムを提案する．

Algorithm 2 RendezvousWithDelta(ASYNC, Non-rigid(+δ), A)

Assumptions: full-light, two colors (A and B)

```
1:  case dis(me.position, other.position)(= DIST) of
2:  DIST > 2δ:
3:      if me.light = other.light = B then
4:          me.des ← the point moving by δ/2 from me.position to other.position
5:      else me.light ← B
6:  2δ ≥ DIST ≥ δ:
7:      if me.light = other.light = A then
8:          me.light ← B
9:          me.des ← the midpoint of me.position and other.position
10:     else me.light ← A
11: δ > DIST: //Rendezvous(ASYNC, Rigid, A)
12:     case me.light of
13:     A:
14:         if other.light = A then
15:             me.light ← B
16:             me.des ← the midpoint of me.position and other.position
17:         else me.des ← other.position
18:     B:
19:         if other.light = A then
20:             me.des ← me.position //stay
21:         else me.light ← A
22:     end case
23: end case
```

2 台間の距離をDISTとする．初期状態を A とし，DIST > 2δの場合，ロボットは互いの状態が B でない限り移動しない．互いの状態 B を観測すると，DISTを$\delta/2$だけ減らすように移動する．2δ ≥ DIST ≥ δの場合，互いの状態 A を観測すると，2 台の中点に移動する．そしてDIST < δの場合，Algorithm1 を利用することで集合できる．

[1] ごく最近，[12]によって肯定的に解決された．

定理７　Algorithm2 について，
RendezvousWithDelta(ASYNC, Non-rigid(+δ), A)はランデブー問題を正しく解く．

５．external-light によるランデブー問題

（１）SSYNC の最適状態数について

[9]より，SSYNC, external-light, Non-rigid，状態数 3 でランデブー問題を解く，クラス$\mathcal{L}$に属するアルゴリズムが知られている．そのアルゴリズムの状態遷移図を図 2 に示す．

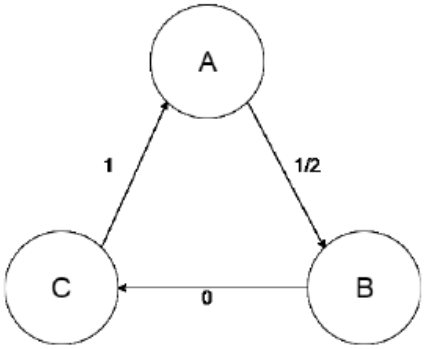

図 2．SSYNC, external-light, 3 状態における状態遷移図
相手がノードの状態を示しているとき，自身は矢印が指し示す状態に遷移する．その際，矢印のラベルが示す移動距離を計算する．

ランデブー問題を解くためには，少なくとも「2 台の中点に移動」，「相手の位置に移動」，「その場で停止」の 3 つの移動パターンが必要である．よって次の定理が成り立つ．

定理８　SSYNC, external-light, Rigid でランデブー問題を解くクラス$\mathcal{L}$のアルゴリズムについて，少なくとも状態数 3 が必要である．

（２）LC-atomic ASYNC，Rigid の場合

[]より，ASYNC, external-light の場合はクラス$\mathcal{L}$に属するランデブー問題を解くアルゴリズムは存在しない．しかし LC-atomic ASYNC を用いることで，クラス$\mathcal{L}$に属するアルゴリズムが存在する．

まず，Rigid の場合，状態数 3 でランデブー問題を解くアルゴリズムを以下に示す．

Algorithm3 は初期状態を A とする．また定理８より，この状態数は最適数となる．このアルゴリズムの状態遷移図を図 3 に示す．

定理９　Algorithm3 について，
RendezvousWithExternal-lightAndRigid(LC-atomic ASYNC, Rigid, A)はランデブー問題を正しく解く．

Algorithm 3 RendezvousWithExternal-lightAndRigid(LC-atomic ASYNC, Rigid, A)

Assumptions: external-light, 4 colors (A, B, and C)

```
case other.light of
A:
    me.light ← B
    me.des ← the midpoint of me.position and other.position
B:
    me.light ← C
    me.des ← me.position //stay
C:
    me.light ← B
    me.des ← other.position
end case
```

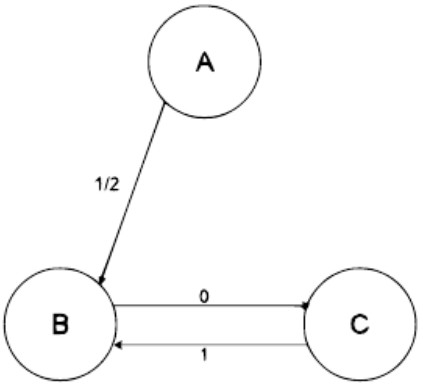

図 3．Algorithm3 の状態遷移図

（３）LC-atomic ASYNC，Non-rigid の場合

Non-rigid とした場合，状態数 4 でランデブー問題を解く準自己安定なアルゴリズムが存在することを示す．

まず，Non-rigid，3 状態でランデブー問題を解くアルゴリズム，Rigid，3 状態でランデブー問題を解く準自己安定なアルゴリズムは存在しないことを示す．

定理１０　LC-atomic ASYNC, external-light, Non-rigid, 3 状態でランデブー問題を解くクラス$\mathcal{L}$のアルゴリズムについて，Rigid の場合，準自己安定なアルゴリズムは存在しない．また Non-rigid の場合，初期状態を 1 つに制限したとしても正しく動作するアルゴリズムは存在しない．

次に，LC-atomic ASYNC, external-light, Non-rigid，状態数 4 でランデブー問題を解くクラス$\mathcal{L}$のアルゴリズムについて，任意の初期状態から開始しても正しく動作するものはないことを示す．

定理１１　LC-atomic ASYNC, external-light, Non-rigid，状態数 4 でランデブー問題を解くクラス$\mathcal{L}$に属する自己安定なアルゴリズムは存在しない．

LC-atomic ASYNC, external-light, Non-rigid，状態数 4 で

ランデブー問題を解く準自己安定なアルゴリズムを以下に示す.

Algorithm 4 RendezvousWithExternal-light(LC-atomic ASYNC,Non-rigid, qS)

Assumptions: external-light, 4 colors (A, B, C, and D)
1: **case** other.light **of**
2: A:
3: $me.light \leftarrow B$
4: $me.des \leftarrow$ the midpoint of $me.position$ and $other.position$
5: B:
6: $me.light \leftarrow C$
7: $me.des \leftarrow me.position$ //stay
8: C:
9: $me.light \leftarrow D$
10: $me.des \leftarrow other.position$
11: D:
12: $me.light \leftarrow A$
13: $me.des \leftarrow me.position$ //stay
14: **end case**

Algorithm4 は準自己安定なアルゴリズムである．そのため，2 台の初期状態が同じであれば正しく動作するが，初期状態が A かつ C，または B かつ D である場合は正しく動作しない．このアルゴリズムの状態遷移図を図 4 に示す.

定理１２ Algorithm4 について，RendezvousWithExternal-light(LC-atomic ASYNC, Non-rigid, *qS*)はランデブー問題を正しく解く.

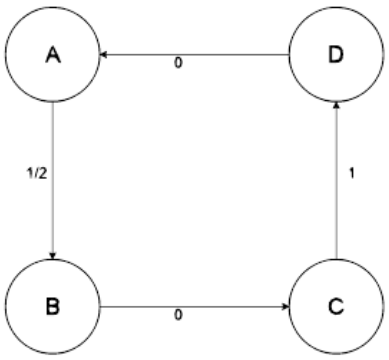

図 4．Algorithm4 の状態遷移図

（４）LC-atomic ASYNC，自己安定の場合

定理１１より，LC-atomic ASYNC，external-light, Non-rigid，状態数 4 でランデブー問題を解くクラス$\mathcal{L}$に属する自己安定なアルゴリズムは存在しない．そこで状態数 5 を使用すれば，ランデブー問題を解く自己安定なアルゴリズムが存在することを示す．そのアルゴリズムを以下に示す.

Algorithm 5 RendezvousWithExternal-lightAndSelfStability(LC-atomic ASYNC,Non-rigid, SS)

Assumptions: external-light, 4 colors (A, B, C, D, and E)
1: **case** other.light **of**
2: A:
3: $me.light \leftarrow B$
4: $me.des \leftarrow$ the midpoint of $me.position$ and $other.position$
5: B:
6: $me.light \leftarrow C$
7: $me.des \leftarrow me.position$ //stay
8: C:
9: $me.light \leftarrow D$
10: $me.des \leftarrow other.position$
11: D:
12: $me.light \leftarrow E$
13: $me.des \leftarrow me.position$ //stay
14: E:
15: $me.light \leftarrow A$
16: $me.des \leftarrow me.position$ //stay
17: **end case**

Algorithm5 は, Algorithm4 にさらに状態を 1 つ追加したものとなる．これによって状態の関係が非対称となるため，任意の初期状態から開始しても正しく動作する．このアルゴリズムの状態遷移図を図 5 に示す.

定理１３ Algorithm5 について，RendezvousWithExternal-lightAndSelfStability(LC-atomic ASYNC, Non-rigid, *SS*)はランデブー問題を正しく解く.

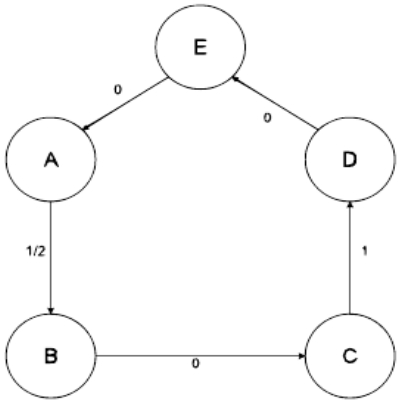

図 5．Algorithm5 の状態遷移図

６．結果と今後の課題

本研究では，状態を持つ自律分散ロボットについて，LC-atomic ASYNC, full-light, Non-rigid，状態数 2 でランデブー問題を解くアルゴリズムと，ASYNC，full-light. Non-rigid(+δ)，状態数 2 でランデブー問題を解くアルゴリズムを提案した．LC-atomic ASYNC，external-light について，Rigid ならば状態数 3 でランデブー問題を正しく解くアルゴリズムを提案した．Non-rigid の場合，状態数 4 で問題を解く準自己安定なアルゴリズム，状態数 5 で問題を解く自己安定なアルゴリズムを提案した．また，SSYNC, external-light の場合，ランデブー問題を正しく解

くには状態数 3 が必要であることを証明した(表 2). 今後の研究課題としては，ASYNC, internal-light，または LC-atomic ASYNC, internal-light の条件でランデブー問題を解くアルゴリズムについての考察や, 3 台以上のロボットで集合問題を解くアルゴリズムについての考察が挙げられる.

表 2. 今回得られた結果[2]

		full	external	internal	−
FSYNC	NR				○
SSYNC	(+δ)			3	×
	R			6	
	NR	2	3	∞	
LC-atomic ASYNC	(+δ)			?	×
	R		**3**	?	
	NR	**2**	**4~5**	?	
ASYNC	(+δ)	**2**	3	?	×
	R	2	12	?	
	NR	2~3	∞	?	

太字で示した部分が本稿で新たに得られた結果である.

謝辞

本研究を行うにあたり，多大なご指導を賜りました和田幸一教授，ならびに名古屋工業大学院工学研究科情報工学専攻片山喜章教授に深く感謝致します．また日常の議論を通じて多くの知識や示唆を頂いた和田研究室の皆様に感謝いたします.

参考文献

1) Noa Agmon, David Peleg ： Fault-tolerant gathering algorithms for autonomous mobile robots, SIAM Journal on Computing, 36, pp.56-82, 2006
2) Zohir Bouzid, Shantanu Das, Sebastien Tixeuil：Gathering of mobile robots tolerating multiple crash Faults. In Proceedings of 33rd IEEE International Conference on Distributed Computing SystemsICDCS, 2013
3) Mark Cieliebak, Paola Flocchini, Giuseppe Prencipe, Nicola Santoro：Distributed computing by mobile robots: Gathering, SIAM Journal on Computing, 41(4): pp.829-879, 2012
4) Shantanu Das, Paola Flocchini, Giuseppe Prencipe, Nicola Santoro, Masafumi Yamashita：Autonomous mobile robots with lights, Theoretical Computer Science, Volume 609, Part 1, pp.171-184, 2015
5) Xavier Defago, Maria Gradinariu Potop-Butucaru, Julien Clement, Stephane Messika, Philippe Raipin Parvedy：Fault and Byzantine tolerant self-stabilizing mobile robots gathering - Feasibility study -. CoRRabs/1602.05546, 2016
6) Bastian Degener, Barbara Kempkes, Tobias Langner, Friedhelm Meyer auf der Heide, Peter Pietrzyk, Roger Wattenhofer：A tight runtime bound for synchronous gathering of autonomous robots with limited visibility, In Proceedings of 23rd ACM Symposium on Parallelism in Algorithms and Architectures (SPAA), pp.139-148, 2011
7) Yoann Dieudonne, Franck. Petit：Self-stabilizing gathering with strong multiplicity detection, Theoretical Computer Science, 428(13), 2012
8) Mattia D'Emidio, Daniele Frigioni, Alfredo Navarra ： Characterizing the Computational Power of Anonymous Mobile Robots, Proceedings of the 29th ACM SIGACT-SIGOPS symposium on Principles of distributed computing, pp.267-276, 2010
9) Paola Flocchini, Nicola Santoro, Giovanni Viglietta, Masafumi Yamashita：Rendezvous of Two Robots with Constant Memory, 20th International Colloquium on Structural Information and Communication Complexity Structural Information and Communication Complexity, Lecture Notes in Computer Science 8179, pp189-200, 2013
10) Paola Flocchini, Giuseppe Prencipe, Nicola Santoro：Distributed Computing by Oblivious Mobile Robots, SYNTHESIS LECTURES ON DISTRIBUTED COMPUTING THEORY, Morgan & Claypool, 2012
11) Paola Flocchini, Giuseppe Prencipe, Nicola Santoro, Peter Widmayer ： Gathering of asynchronous robots with limited visibility, Theoretical Computer Science, 337(1-3): pp.147-168, 2005
12) Adam Heriban, Xavier Defago, Sebastien Tixeuil：Optimally Gathering Two Robots, Research Report HAL Id: hal-01575451, UPMC Sorbonne Universites, Aug. 2017
13) Taisuke Izumi, Zohir Bouzid, Sebastien Tixeuil, Koichi Wada, Brief Announcement: The BG-simulation for Byzantine mobile robots, 25th DISC, pp.330-331, 2011
14) Taisuke Izumi, Yoshiaki Katayama, Nobuhiro Inuzuka, Koichi Wada ： A Difference-Optimal Algorithm for Gathering Autonomous Mobile Robots with Dynamic Compasses, An Optimal Result, 21st International Symposium on Distributed Computing (DISC 2007), Lecture note in Computer Science 4731, pp 298-312, 2007
15) Taisuke Izumi, Souissi Souissi, Yoshiaki Katayama, Nobuhiro Inuzuka, Xavier Defago, Koichi Wada, Masafumi Yamashita, The gathering problem for two oblivious robots with unreliable compasses, SIAM Journal on Computing, 41(1):, pp.26-46, 2012
16) Giuseppe Antonio Di Luna, Paola Flocchini, Sruti Gan Chaudhuri, Nicola Santoro, Giovanni Viglietta ： Robots with Lights: Overcoming Obstructed Visibility Without Colliding, Stabiliztion, Safety, Security of Distributed System pp.150-164, 2014
17) Linda Pagli, Giuseppe Prencipe, Giovanni

[2] ASYNC, full-light, Non-rigid, 状態数 2 のアルゴリズムについては[12]によって証明されたものである.

18) Viglietta : Getting Close Without Touching: Near-Gathering for Autonomous Mobile Robots, Distributed Computing, Volume 28, Issue 5, pp.333-349, 2015
19) Samia Souissi, Xavier Defago, Masafumi Yamashita : Using eventually consistent compasses to gather memory-less mobile robots with limited visibility, ACM Transactions on Autonomous and Adaptive Systems, 4(1): pp.1-27, 2009
20) Satoshi Terai, Koichi Wada, Yoshiaki Katayama, Gathering problems for autonomous mobile robots with lights, Technical Report of Wada Labo, Hosei University, 2016
21) Giovanni Viglietta : Rendezvous of two robots with visible bits, Technical Report arXiv:1211.6039, 2012
22) Masafumi Yamashita, Ichiro Suzuki : Characterizing geometric patterns formable by oblivious anonymous mobile robots, Theoretical Computer Science Volume 411 Issue 26-28, pp.2433-2453, 2010

この論文に関する発表

1) Takashi Okumura, Koichi Wada, Yoshiaki Katayama : Rendezvous of Autonomous Mobile Robots with Lights in Asynchronous Schedulers, The 20th Korea-Japan Joint Workshop on Algorithms and Computation, 2017
2) 奥村太加志，和田幸一，片山喜章 : ライトを持つ 2 台の自律分散ロボットのランデブーについて, 第 13 回情報科学ワークショップ，2017
3) Takashi Okumura, Koichi Wada, Yoshiaki Katayama : Optimal Asynchronous Rendezvous for Mobile Robots with Lights, 19th International Symposium on Stabilization, Safety, and Security of Distributed Systems, 2017