論　文

自律移動ロボット群のための停止故障耐性のある分散型問題解法

吉田　大輔[†]　　増澤　利光[††]　　藤原　秀雄[††]

Fault Tolerant Distributed Algorithms for Autonomous Mobile Robots with Crash Faults

Daisuke YOSHIDA[†], Toshimitsu MASUZAWA[††], and Hideo FUJIWARA[††]

あらまし　自律移動ロボット群が協調作業を行うための分散アルゴリズムについて考える．ロボットは，同じプログラムを非同期に実行し，視覚センサによって得られた他のロボットの位置から，自分の動作を決定するものとする．本論文では，初期停止故障ロボットが混在するロボット群について考察する．まず，正常なロボットのみからなるグループを構成する問題を正常ロボット選出問題として定義し，過半数のロボットが正常なときに，この問題を解くアルゴリズムを示す．そして，正常ロボット選出問題の解を利用することにより，故障耐性のないアルゴリズムを故障耐性のあるアルゴリズムに変換する方法について考察する．

キーワード　自律移動ロボット，分散アルゴリズム，故障耐性，停止故障アルゴリズム変換

1. まえがき

平面上に配置された多数の自律移動ロボットが協調して作業する状況を考える．自律移動ロボットは，宇宙空間での任務，危険な環境下での作業など，将来的にさまざまな分野での活躍が期待されている．複数のロボットの動作を制御する方法として，コントロールセンタがすべてのロボットの状態を把握し，そこで集中的に動作を制御するという集中制御型解法も考えられる．しかし，通常のシステムと同様，処理性能，拡張性，信頼性などの点から，各ロボットが自分の得た情報から自らの動作を決定する分散型解法が望ましい．これまでに，このような自律移動ロボット群による分散型問題解法に関する研究はいくつか行われているが，マイクロロボット技術の進歩などによって，今後ますます重要になってくると思われる．

文献[1],[3]～[6]では，以下の仮定のもとで，自律移動ロボット群の協調動作に関する研究を行っている．

(i)　ロボットは大きさをもたない点ロボットである．

(ii)　すべてのロボットは，識別子をもたず，同じプログラムを実行する．また，外観などによって，他のロボットと区別することはできない．

(iii)　ロボットは通信機能をもたず，視覚センサによって得られた他のロボットの位置から，自分の次の動作を決定する．但し，各ロボットは，独自のローカル直交座標系をもち，自分や他のロボットの位置をこのローカル直交座標系で認識している．ロボットによって，そのローカル直交座標系の原点の位置，x 軸の方向，単位距離は異なる．

(iv)　ロボットは，非同期に動作する（[1]以外）．つまり，動作の速いロボットや，遅いロボットが混在している．

(v)　ロボットは，瞬時に移動できる．

文献[3]では，すべてのロボットを円状に整列する問題などについて，アルゴリズムを提案し，その性能をシミュレーションによって評価している．また，文献[4],[5]では，ロボットを1点に集める問題や，すべてのロボットに共通の原点や単位距離を認識させるという一種の合意問題について，そのアルゴリズムや非可解性を理論的に議論している．これらの文献では，各ロボットは視覚センサによって他のすべてのロボットの位置を知ることができると仮定している．一方，文献[6]では，視覚センサの有効距離に制限をおいた場合の円状整列問題についてアルゴリズムを提案し，そ

† (株)日立製作所汎用コンピューター事業部，秦野市
General Purpose Computer Division, Hitachi Ltd., Hadano-shi, 259-13 Japan
†† 奈良先端科学技術大学院大学情報科学研究科，生駒市
Graduate School of Information Science, Nara Institute of Science and Technology, Ikoma-shi, 630-01 Japan

の性能をシミュレーションによって評価している．また，文献[1]では，視覚センサに同様の制限をおいた場合について理論的な検討を加えている．

これらの研究では，いずれもすべてのロボットは正常で，プログラムを正しく実行すると仮定している．しかし，ロボットの数が多くなると，ロボット群に故障ロボットが含まれる状況は避けられない．従って，故障ロボットが存在するときに，正常ロボットが協調作業するためのアルゴリズムを設計することは，通常の分散システムの場合と同様，非常に重要である．

そこで，故障ロボットが混在するロボット群でのアルゴリズムについて考察する．自律分散移動ロボット群のための故障耐性のあるアルゴリズムの第1ステップとして，本論文では，初期停止故障のみを考慮する．初期停止故障とは，故障ロボットは一切動作せず，また，アルゴリズム実行開始時から故障している（つまり，アルゴリズム実行中に新たな故障は起こらない）という故障のモデルである．

非同期式ロボット群では，停止故障ロボットと，単に動作の遅い正常なロボットを有限時間内に区別することができない．そのため，停止故障は，故障が非同期式システムに及ぼす影響の本質的な一面を特化させたものと考えられる．また，計算機がネットワークで接続された通常の分散システムにおいて，アルゴリズム実行中に1台の計算機でも停止故障を起こすと，合意問題のような基本的な問題が解けないことが知られている[2]．そこで，ここではアルゴリズム実行中に生じる停止故障は考慮せず，初期停止故障のみを扱う．

初期停止故障ロボットが混在するロボット群で，協調作業を行う一つの方法として，正常なロボットのみからなるグループを選び出し，そのグループに属するロボットが互いにグループ内の他のロボットを認識することにより，そのグループ内のロボットだけで協調して問題を解くという方法が考えられる．先に述べたように，非同期式ロボット群においては初期停止故障ロボットを検出することはできない．しかし，一度でも動作したロボットは正常ロボットであると判断できるので，正常ロボット（の一部）からなるグループを構成することは不可能ではない．そこで，正常なロボットのみからなるグループを構成し，そのグループを構成するロボットが，互いにグループ内の他のロボットを認識するという問題を正常ロボット選出問題として定義する．そして，過半数のロボットが正常のときに，この問題を解くアルゴリズムを示す．そして，正常ロボット選出問題の解を利用することにより，故障耐性のないアルゴリズムを故障耐性のあるアルゴリズムに変換する方法について考察する．

以下，**2.**では，ロボット群に関するいくつかの定義を行う．**3.**では，正常ロボット選出問題を定義し，それを解くアルゴリズムを示す．**4.**では，正常ロボット選出問題の解を利用することにより，故障耐性のないアルゴリズムを故障耐性のあるアルゴリズムに変換する方法について考察する．**5.**では，結論を述べる．

2. 諸 定 義

n 台の点ロボット $\mathcal{N}=\{r_1, r_2, \cdots, r_n\}$ が，x-y 直交座標系をもつ平面上に存在する状況を考える．この直交座標系を絶対座標系 Z と呼び，Z 上でのロボット r の位置座標 (x, y)（x, y は実数）をロボット r の絶対座標と呼ぶ．各ロボット r は絶対座標系 Z を知らず，r 独自の x-y 直交座標系 Z_r（ローカル座標系と呼ぶ）をもつ．Z_r の原点，単位距離，座標軸の方向は，絶対座標 Z や他のロボット q のローカル座標系 Z_q の原点，単位距離，座標軸の方向とそれぞれ一致するとは限らない．つまり，ある定数 $d_{x,r}$，$d_{y,r}$，θ_r，a_r（但し，$a_r \neq 0$）が存在し，絶対座標 Z 上の任意の点 (x, y) は次式によって定まる Z_r 上の点 (x', y') に一致する．但し，ロボット r は，$d_{x,r}$，$d_{y,r}$，θ_r，a_r を知らず，これらの値をアルゴリズムでは利用できないものとする．

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = a_r \begin{bmatrix} \cos\theta_r & -\sin\theta_r \\ \sin\theta_r & \cos\theta_r \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} + \begin{bmatrix} d_{x,r} \\ d_{y,r} \end{bmatrix}$$

［定義1］（ロボット） 各ロボット r は，ローカル座標系 Z_r での自分の位置を知っている．また，視覚センサをもち，他のすべてのロボットのその瞬間での Z_r での位置を，座標の多重集合(注1)として知ることができる．ここで，他のロボットの位置情報が座標の（系列ではなく）多重集合として得られることは，r が他のロボットを区別できないことを表している．

各ロボット r は，平面上を移動可能な決定性（無限）状態機械 (Q, L, ψ, q_0, Fin) で定義される．ここで，Q，L，ψ，q_0，Fin は，それぞれ，(無限) 状態集合，Z_r での座標の（無限）集合，状態遷移関数，初期状態 $(q_0 \in Q)$，最終状態の集合 $(Fin \subseteq Q)$ を表す．状態遷移関数 $\psi : Q \times L \times \{L^{n-1}\} \to Q \times L$（ここ

(注1)：マルチセット．同じ値の要素を複数もつことを許した集合．

で，$\{L^{n-1}\}$ は $n-1$ 個の座標からなる多重集合の集合を表す）は，ロボット r の状態と Z_r での位置，センサで得られた他のすべてのロボットの Z_r での位置から，r の新たな状態と移動先を決定する．但し，最終状態からは状態遷移も移動も起こらない（最終状態はプログラムの終了状態に対応する）． □

ロボット群のためのアルゴリズムとは，各ロボットに対応する状態機械（各ロボットが実行するプログラム）を定めるものである．以下の仮定をおく．

[仮定1] すべてのロボットは同じ状態機械である（識別子をもたず，同じプログラムを実行する）． □

[仮定2] 視覚センサによる位置情報の獲得，状態遷移，移動の一連の動作は，原子動作であり，瞬時に実行できる． □

本論文では，故障ロボットを含むロボット群のためのアルゴリズムを考える．故障のモデルとしてはさまざまなモデルが考えられるが，すべての故障ロボットは初期状況において既に故障しており（つまり，アルゴリズム実行中に新たな故障は生じない），故障ロボットは一切動作しないと仮定する．このような故障を，初期停止故障と言う（以下では，単に故障と言う）．また，故障していないロボットを正常ロボットと呼ぶ．以下では，故障ロボットの集合を $\mathcal{F}$，正常ロボットの集合を $\mathcal{C}(=\mathcal{N}\backslash\mathcal{F})$ と表す．

[定義2] （大域状況） ロボット群 $\mathcal{N}$ の大域状況は，すべてのロボットの状態，絶対座標系 Z での位置からなる．但し，故障ロボットを表すために，ロボットの状態として新たに故障状態 $f(\notin Q)$ を導入する．つまり，ロボット群 $\mathcal{N}=\{r_1,r_2,\cdots,r_n\}$ の大域状況を $\langle\alpha_1,\beta_1,\alpha_2,\beta_2,\cdots,\alpha_n,\beta_n\rangle$ で表す．ここで，各 $i(1\leq i\leq n)$ について，r_i が正常ロボットなら α_i は r_i の（f 以外の）状態を表し，r_i が故障ロボットなら $\alpha_i=f$ である．また，β_i は r_i の Z での座標を表す．特に，すべてのロボットの状態が初期状態または故障状態である大域状況を初期大域状況と言い，各ロボットの初期大域状況での位置を初期位置と言う．また，すべてのロボットの状態が最終状態または故障状態である大域状況を最終大域状況と言う． □

本論文では，初期大域状況でのロボットの配置について，次のように仮定する．

[仮定3] すべてのロボットの初期位置は互いに異なる． □

[定義3] （イベント） ロボットの原子動作の実行をイベントと言う．ロボット r のイベントは $\langle r,\alpha,\beta,\alpha',\beta'\rangle$ で表される．ここで α,β はそれぞれイベント実行前の r の状態，位置を表し，α',β' はそれぞれイベント実行後の r の状態，位置を表す．イベントには，位置情報の獲得，状態遷移だけでロボットが移動しないイベントもあるが，特に，ロボットが移動するイベントを移動イベントと言う．大域状況 γ において，あるロボット r がイベント ϵ を起こすことが可能であるとき，ϵ を γ に適用可能なイベントと言う． □

大域状況 γ において最終状態または故障状態にあるロボット r について，γ に適用可能な r のイベントは存在しない．最終状態でも故障状態でもないロボット r について，γ に適用可能な r のイベントは（r が決定性状態機械なので）一つしか存在しない．

大域状況 γ で適用可能なすべてのイベントの集合を $\mathcal{E}(\gamma)$ とする．$\mathcal{E}(\gamma)$ の空でない部分集合 $\mathcal{E}'(\gamma)$ のすべてのイベントが同時に起こることにより，γ は別の大域状況 δ に変化する．このとき，$\gamma\rightarrow_{\mathcal{E}'(\gamma)}\delta$ と表す．つまり，$\gamma=\langle\alpha_1,\beta_1,\alpha_2,\beta_2,\cdots,\alpha_n,\beta_n\rangle$，$\delta=\langle\alpha_1',\beta_1',\alpha_2',\beta_2',\cdots,\alpha_n',\beta_n'\rangle$ とするとき，各 $i(1\leq i\leq n)$ について，r_i のイベント ϵ_i が $\mathcal{E}'(\gamma)$ に属するならば，α_i',β_i' は ϵ_i 後の r_i の状態，位置を表す．r_i のイベントが $\mathcal{E}'(\gamma)$ に属さないならば，$\alpha_i'=\alpha_i$,$\beta_i'=\beta_i$ である．

[定義4] （実行） アルゴリズムの実行は，大域状況の（有限または無限）列 $\gamma_0,\gamma_1,\gamma_2,\cdots$ として表される．ここで，γ_0 は初期大域状況であり，実行が有限列 $\gamma_0,\gamma_1,\gamma_2,\cdots,\gamma_f$ であるならば，γ_f は最終大域状況である．また，各 $j(j\geq 0)$ について，γ_j で適用可能なイベントの空でない（部分）集合 $\mathcal{E}'(\gamma_j)$ が存在し，$\gamma_j\rightarrow_{\mathcal{E}'(\gamma_j)}\gamma_{j+1}$ が成り立つ． □

実行の定義において，γ_j で適用可能なイベント $\mathcal{E}(\gamma_j)$ の空でない部分集合 $\mathcal{E}'(\gamma_j)$ のすべてのイベントが同時に起こるということは，各ロボットは非同期に動作する，すなわち，ロボットの動作速度に関する仮定がないことをモデル化している．このことにより，大域状況 γ において，$\mathcal{E}(\gamma)$ のうちどのイベントが γ で起きるかはわからない．従って，一般に，同じ初期大域状況から始まる実行は複数存在する．但し，動作可能な正常ロボットは，いつか必ず動作すると仮定する．

[仮定4] 任意の実行の任意の大域状況を γ とする．γ において，最終状態でも故障状態でもない任意のロボット r について，γ 以降に r のイベントは少なくとも1回起こる． □

[定義5] (問題) 分散問題は，実行に関する条件で定義される．分散問題 P を定義する実行に関する条件を Ψ^P と表す．k を非負整数，A をアルゴリズムとする．このとき，$|\mathcal{F}| \leq k$ を満たす任意の初期大域状況から始まる A の任意の実行が条件 Ψ^P を満たすならば，アルゴリズム A は k 個の故障にかかわらず問題 P を解くと言う． □

3. 正常ロボット選出問題

3.1 問題の定義

故障ロボットを含むロボット群において，故障ロボットを検出できれば，正常なロボットだけが協調して問題を解くことにより，故障耐性のあるアルゴリズムを構成できる．しかし，非同期システムにおいては，あるロボット r が移動しないとき，r が故障ロボットであるのか，動作の遅い正常ロボットであるのかを有限時間内に区別することはできない．しかし，一度でも移動したロボットは正常ロボットであると判断できるので，正常ロボットの選出は可能である．そこでこの節では，故障ロボット数の上界 k を仮定し，$n-k$ 台以上の正常ロボットからなる集合 $\mathcal{M}(\subseteq\mathcal{C})$ を求める問題について考察する．但し，$n-k$ 台以上からなる互いに素な正常ロボット集合が異なる集合を求めないために，ここでは，$0 \leq k < \lceil n/2 \rceil$ とする．

[定義6] (正常ロボット選出問題) 正常ロボット選出問題を解く任意の実行において，ある正常ロボットの集合 $\mathcal{M}(\subseteq\mathcal{C}, |\mathcal{M}| \geq n-k)$ が存在し，任意のロボット $r(\in \mathcal{M})$ は $\mathcal{M}$ を求めて最終状態に遷移する．任意のロボット $q(\in \mathcal{C}\backslash\mathcal{M})$ は，最終状態に遷移するならば $q \notin \mathcal{M}$ であることを知る (q は最終状態に遷移しなくてもよい(注2))．このロボット集合 $\mathcal{M}$ を正常ロボット解集合と言う． □

3.2 アルゴリズム *A_Select*

ここでは，k $(0 \leq k < \lceil n/2 \rceil)$ 個の故障にかかわらず正常ロボット選出問題を解くアルゴリズム *A_Select* (*Active_Select*) を示す．

アルゴリズムの説明，および，正当性の証明のために，すべてのロボットに共通な絶対時刻を導入する(注3)．ロボット r が $i(\geq 1)$ 回目のイベントを実行する絶対時刻を $E_r(i)$，このとき視覚センサにより得られる他のすべてのロボットの位置情報を $P_r(i)$，$t(\geq 1)$ 回目の移動イベントを実行する絶対時刻を $T_r(t)$ と表す．但し，$T_r(0) = 0$ とし，r が t_r 回しか移動イベントを実行しないとき，任意の $t'(> t_r)$ について $T_r(t') = \infty$ とする．$E_r(i)$ についても同様にする．また，$\{x | a \leq x < b\}$ なる絶対時刻の集合（期間）を $[a, b)$ と表す．

$P_r(i)$,$P_r(i+1)$ $(i \geq 1)$ はそれぞれ $n-1$ 個の座標からなる多重集合であるが，$P_r(i)$,$P_r(i+1)$ $(i \geq 1)$ の間で同じロボットの座標を対応づけることは一般には不可能である．つまり，各座標 $v \in P_r(i)$ に対し，時刻 $E_r(i)$ に v に位置していたロボットの，時刻 $E_r(i+1)$ での位置 $v' \in P_r(i+1)$ を決定することはできない．しかし，アルゴリズムによってはロボットの移動の規則性などから，対応付けが可能である場合がある．そのようなアルゴリズムを追跡可能なアルゴリズムと言う．アルゴリズム *A_Select* では各ロボットが移動する範囲を制限することで追跡可能としている（補題2）．従って *A_Select* では，各ロボットは他のすべてのロボットを区別可能としている．

[アルゴリズム *A_Select* (各ロボット r の動作)]

(変数)

ct_r：移動回数を表すカウンタ（初期値は0）．

d_r：移動距離を表す変数．

$flag_{A\_Select}$：ループの脱出を判定する論理型変数 (初期値は *true*)．

MR_r：$ct_r = t$ のとき，ロボットの集合 $\{p | T_r(t)$ 以降に p の移動イベントを r が観測した$\}$（初期値は ϕ）．

$preMR_r$：$ct_r = t$ のとき，ロボットの集合 $\{p | [T_r(t-1), T_r(t))$ に p の移動イベントを r が観測した$\}$．

L_r：ロボットの集合 $\{p | p$ が左折したことを r が観測した$\}$（初期値は ϕ）．

(動作)

一般性を失うことなくローカル座標系 Z_r での r の初期位置を $(0,0)$ と仮定する．また，以下の動作において，移動先の座標はすべて Z_r における座標である．

(a) $ct_r = 0$ のとき（起動時）：$P_r(1)$ において，r に最も近いロボットと r との距離を 1/16 倍した距離 d_r を求める．$(d_r, 0)$ に移動し，$ct_r := 1, MR_r := \{r\}$

(注2)：$\mathcal{M}$ に属するすべてのロボットがアルゴリズムの実行を終了した後に始動するロボット r が存在するかもしれない．このロボット r は一般には $\mathcal{M}$ が既に求まっているのかどうかを知ることはできないため，他のロボットが動くのを永久に待つかもしれない．このとき，r はアルゴリズムを終了できない．このようなロボット r を最終状態に遷移させるには，例えば，ある2台以上の $\mathcal{M}$ のロボットが同じ位置に移動して終了するようにすれば仮定3より可能である．しかし，本論文では $\mathcal{M}$ を他の問題を解くことに利用するので，$\mathcal{M}$ のすべてのロボットが異なる座標に位置する状況で停止させたい．そのため，$\mathcal{M}$ に属さない正常ロボットについては，最終状態に遷移しないことを許している．

(注3)：絶対時刻は説明，および，正当性の証明のためだけに導入するものであり，各ロボットが実行するプログラムでは利用できない．

を実行する．

(b) $ct_r = 1$ のとき：$|MR_r| \geq n - k$ が成り立つまで，視覚センサによる観測を繰り返す．このとき，ロボット p の移動を観測すると，p を MR_r に加える．$|MR_r| \geq n - k$ になれば，$(2d_r, 0)$ に移動し，$ct_r := 2, preMR_r := MR_r, MR_r := \{r\}$ を実行する．

(c) $ct_r = 2$ のとき：$MR_r \supseteq preMR_r$ が成り立つまで，視覚センサによる観測を繰り返す．このとき，ロボット p の移動を観測すると，p を MR_r に加える．$MR_r \supseteq preMR_r$ になれば，$(3d_r, 0)$ に移動し，$ct_r := 3, preMR_r := MR_r, MR_r := \{r\}$ を実行する．

(d) $ct_r = 3$ のとき：$MR_r \supseteq preMR_r$ が成り立つまで，視覚センサによる観測を繰り返す．このとき，ロボット p の移動を観測すると，p を MR_r に加える．$MR_r \supseteq preMR_r$ になれば，r は次のいずれかの動作をする．

(d−1) $ct_p = 5$ であるロボット p が存在するならば（r は p の位置の履歴から $ct_p = 5$ かどうかを判定できる（補題7）），$(4d_r, 0)$ に移動し，$ct_r := 4$ を実行する．（このような r を直進ロボット (straight−moved robot) と言う）．このとき，r は r が正常ロボット解集合に属さないことを知って，アルゴリズムを終了する．

(d−2) $ct_p = 5$ であるようなロボット p が存在しないとき，$(3d_r, d_r)$ に移動する（このような r を左折ロボット (left−turned robot) と言う）．$ct_r := 4, preMR_r := MR_r, MR_r := \{r\}$ を実行する．

(e) $ct_r = 4$ のとき：$MR_r \supseteq preMR_r$ が成り立つまで，視覚センサによる観測を繰り返す．このとき，ロボット p の移動を観測すると，p を MR_r に加える．$MR_r \supseteq preMR_r$ になれば，r は初期位置と現在の位置の中点 $(3/2d_r, 1/2d_r)$ に移動し，$ct_r := 5$ を実行する．その後，視覚センサによる観測を1回行い，左折ロボットの集合 L_r（r は各ロボットの位置の履歴から左折ロボットであるかどうかを判定できる（補題6））を解集合とし，アルゴリズムを終了する． □

ロボットの移動の様子を図1に，アルゴリズム *A_Select* の詳細（ロボット r のプログラム）を図2に示す．プログラム中の手続き *watch* は，視覚センサで他のロボットの位置情報を知り，位置の履歴に追加するための手続きである．

3.3 アルゴリズム *A_Select* の正当性

アルゴリズム *A_Select* が k 個の故障にかかわらず正常ロボット選出問題を解くことを示す．以下では，ロボット $r \in \mathcal{N}$ の初期位置（絶対座標）を $r(0)$，$t(\geq 1)$

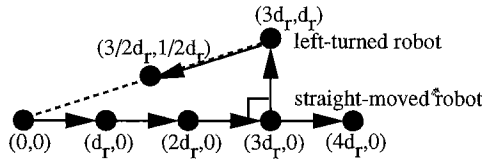


図1 アルゴリズム *A_Select* での r の軌跡
Fig. 1 Locus of robot r in algorithm *A_Select*.

```
A_Select() {
  watch;
  Action_A_Select();
  do {
    Wait_A_Select();
    Action_A_Select();
  } while (flag_A_Select);
}
```

(a) Main Program

```
Wait_A_Select() {
  preMR_r:=MR_r;  MR_r:={r};
  do {
    watch;
    update MR_r; /* Append the robots which have
                    moved after T_r(ct_r) to MR_r */
  } while (MR_r⊂preMR_r∨|MR_r|<n-k);
}
```

(b) Procedure Wait_A_Select

```
Action_A_Select() {
  switch (ct_r) {
    case 0 :
      d_r:=(distance between r and the nearest
            robot on P_r(1))/16;
      move to (d_r,0);  ct_r:=1;
    case 1 :
      move to (2d_r,0); ct_r:=2;
    case 2 :
      move to (3d_r,0); ct_r:=3;
    case 3 :
      if (∃p∈MR_r[ct_p=5]) then {
        move to (4d_r,0);  ct_r:=4;
                           /* straight-moved robot */
        flag_A_Select:=false;
      } else
        move to (3d_r,d_r); ct_r:=4;
                           /* left-turned robot */
    case 4 :
      move to (3d_r/2,d_r/2); ct_r:=5;
      watch;
      L_r:={p|p is left-turned robot};
      flag_A_Select:=false;
  }
}
```

(c) Procedure Action_A_Select

図2 アルゴリズム *A_Select* (ロボット r)
Fig. 2 Algorithm *A_Select* (for robot r).

回目の移動イベント実行後の絶対座標を $r(t)$ とする．

最初に，アルゴリズム *A_Select* は追跡可能であることを示す．*A_Select* では，各ロボット r はすべての $i(\geqq 2)$ に対し，$P_r(i)$ の各座標を $P_r(i-1)$ の中で最も近い座標と対応づける．つまり，$P_r(i)$ で座標 v にあるロボットを $P_r(i-1)$ で以下のように定まる座標 v' にあったロボットとみなす．

$d(v,v') = min\{d(v,w)|w \in P_r(i-1)\}$

($d(v,w)$ は v,w 間のユークリッド距離を表す)

この対応付けが正しいことを示す．これを示すには，任意のロボット $r \in \mathcal{N}$ について，

$max\{d(r(i),r(j))|0 \leqq i < j \leqq 5\} <$
$min\{d(r(i),q(j))|q \in \mathcal{N}\backslash\{r\},$
$0 \leqq i \leqq 5, 0 \leqq j \leqq 5\}$

が成り立つことを証明すればよい．

[補題1] 任意のロボットを r，初期大域状況において r に最も近いロボットと r との距離を D_r とする．任意の実行において，

$max\{d(r(i),r(j))|0 \leqq i < j \leqq 5\} < 1/3D_r$

が成り立つ．

(証明) r が $P_r(1)$ から得る r に最も近いロボット $r'(r \neq r')$ と r との距離を D'_r とする．このとき，r の1回の移動距離は $d_r = 1/16D'_r$ である．r が位置する座標のうちその間の距離が最大であるのは，$r(0)$ と直進ロボットの場合の $r(4)$ であり，その距離は $1/4D'_r$ である（図1）．つまり，$max\{d(r(i),r(j))|0 \leqq i < j \leqq 5\} \leqq 1/4D'_r$ が成り立つ．

初期大域状況において r に最も近いロボットを q とする ($d(r(0),q(0)) = D_r$ である)．

(a) $E_q(1) \leqq E_r(1)$ のとき：$P_q(1)$ において q に最も近いロボット $q'(q \neq q')$ と q との距離を D'_q とすると，$D'_q \leqq D_r$ が成り立つ．$max\{d(q(i),q(j))|0 \leqq i < j \leqq 5\} \leqq 1/4D'_q$ より，$D'_r \leqq D_r + max\{d(q(i),q(j))|0 \leqq i < j \leqq 5\} \leqq D_r + 1/4D'_q \leqq (1+1/4)D_r = 5/4D_r$ が成り立つ．よって，$max\{d(r(i),r(j))|0 \leqq i < j \leqq 5\} \leqq 1/4D'_r \leqq 5/16D_r < 1/3D_r$ が成り立つ．

(b) $E_q(1) > E_r(1)$ のとき：$D'_r \leqq D_r$ が成り立つ．よって，$max\{d(r(i),r(j))|0 \leqq i < j \leqq 5\} \leqq 1/4D'_r \leqq 1/4D_r < 1/3D_r$ が成り立つ． □

[補題2] アルゴリズム *A_Select* は追跡可能である．すなわち，任意のロボット $r \in \mathcal{N}$ について，

$max\{d(r(i),r(j))|0 \leqq i < j \leqq 5\} <$
$min\{d(r(i),q(j))|q \in \mathcal{N}\backslash\{r\},$
$0 \leqq i \leqq 5, 0 \leqq j \leqq 5\}$

が成り立つ．

(証明) 背理法で証明する．あるロボット $r,q(r \neq q)$，整数 i,j,i',j' $(0 \leqq i \leqq 5, 0 \leqq j \leqq 5, 0 \leqq i' < j' \leqq 5)$ が存在し，$d(r(i),q(j)) \leqq d(r(i'),r(j'))$ が成り立つと仮定する．

補題1より $d(r(i'),r(j')) < 1/3D_r$ であるから，$d(r(i),q(j)) < 1/3D_r$ が成り立つ．また $d(r(0),r(i)) < 1/3D_r$，$d(q(0),q(j)) < 1/3D_q$ も成り立つ．よって，$d(r(0),q(0)) \leqq d(r(0),r(i)) + d(r(i),q(j)) + d(q(0),q(j)) < 2/3D_r + 1/3D_q$ が成り立つ．

$D_r \leqq d(r(0),q(0)) < 2/3D_r + 1/3D_q$ より $D_r < D_q$ が成り立つ．同様に，$D_q \leqq d(r(0),q(0)) < 2/3D_r + 1/3D_q$ より $D_q < D_r$ が成り立つ．これらは矛盾する． □

ロボットの移動のための条件より，*A_Select* の任意の実行について，次の二つの性質が成り立つ．

[性質1] 任意のロボットを $r \in \mathcal{C}$ とする．$T_r(2) \neq \infty$ ならば，任意の $t(\geqq 2)$ に対し，$[T_r(1),T_r(t))$ の間に $n-k$ 台以上のロボットが移動イベントを実行する． □

[性質2] 任意のロボット $r \in \mathcal{C}$，任意の $t(\geqq 3)$ に対し $T_r(t) \neq \infty$ ならば，$[T_r(1),T_r(t-1))$ に移動イベントを実行したすべてのロボットは $[T_r(t-1),T_r(t))$ に移動イベントを実行する． □

性質2から次の補題を証明できる．

[補題3] 任意の二つのロボット $r,r' \in \mathcal{C}$，任意の t,t' $(t \geqq 2, t' \geqq 2)$ について，$T_r(t) < T_{r'}(t')$ かつ $T_r(t+1) \neq \infty$ が成り立つならば，$T_r(t+1) < T_{r'}(t'+1)$ が成り立つ．また，$T_r(t) < T_{r'}(t')$ かつ $T_r(t+1) = \infty$ が成り立つならば，$T_{r'}(t'+1) = \infty$ が成り立つ．

(証明) $T_r(t+1) \neq \infty$ の場合について，背理法で証明する．($T_r(t+1) = \infty$ の場合も同様に証明できる．)

$\exists r,r' \in \mathcal{C}, \exists t,t'(\geqq 2)$ $[(T_r(t) < T_{r'}(t')) \wedge (T_r(t+1) \neq \infty) \wedge (T_r(t+1) \geqq T_{r'}(t'+1))]$ が成り立つと仮定する．

(a) $T_{r'}(1) \leqq T_r(t)$ のとき（図3(a)）：$T_r(t) < T_{r'}(t')$ より r は $[T_{r'}(1),T_{r'}(t'))$ に移動イベントを実行している．$T_r(t+1) \neq \infty, T_{r'}(t'+1) \leqq T_r(t+1)$ より $T_{r'}(t'+1) \neq \infty$ である．従って，性質2より r は $[T_{r'}(t'),T_{r'}(t'+1))$ に移動イベントを実行する．これは $T_{r'}(t'+1) \leqq T_r(t+1)$ に矛盾する．

(b) $T_r(t) < T_{r'}(1)$ のとき（図3(b)）：$[T_r(1),T_r(t))$，$[T_{r'}(1),T_{r'}(t'))$ に移動イベントを実行し

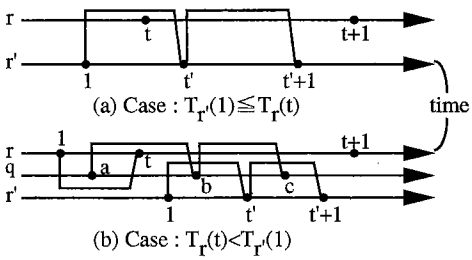


図3 補題3の証明における場合分け
Fig. 3 Cases considered in the proof of Lemma3.

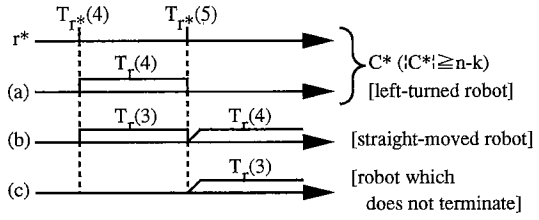


図4 正常ロボット集合 $\mathcal{C}^*$
Fig. 4 Set $\mathcal{C}^*$ of non-faulty robots.

たロボットの集合を，それぞれ，$\mathcal{C}_1$，$\mathcal{C}_2$ とする．性質1より $|\mathcal{C}_1| \geq n-k, |\mathcal{C}_2| \geq n-k$ である．$k < \lceil n/2 \rceil$ より $|\mathcal{C}_1 \cap \mathcal{C}_2| \geq 1$ である．従って，ある $q \in \mathcal{C}_1 \cap \mathcal{C}_2$ が存在し，ある a,b $(1 \leqq a < b)$ に対し $T_r(1) \leqq T_q(a) < T_r(t)$, $T_{r'}(1) \leqq T_q(b) < T_{r'}(t')$ が成り立つ．また性質2より q は $[T_{r'}(t'), T_{r'}(t'+1))$ に移動イベントを実行する．つまり，ある $c(>b)$ に対し $T_{r'}(t') \leqq T_q(c) < T_{r'}(t'+1)$ が成り立つ．r は $[T_q(a), T_q(b))$ に移動イベントを実行しているので，性質2より r は $[T_q(b), T_q(c))$ に移動イベントを実行する．すなわち，$T_r(t+1) < T_q(c)$ である．$T_q(c) < T_{r'}(t'+1)$ より $T_r(t+1) < T_{r'}(t'+1)$ が成り立つが，これは $T_{r'}(t'+1) \leqq T_r(t+1)$ に矛盾する． □

補題3より次の系1，2が成り立つ．

[系1] 任意の二つのロボット $r, r' \in \mathcal{C}$，任意の t, t' $(t \geqq 3, t' \geqq 3)$ について，$T_r(t) < T_{r'}(t')$ が成り立つならば，任意の $u(1 \leqq u \leqq min\{t-2, t'-2\})$ について $T_r(t-u) \leqq T_{r'}(t'-u)$ が成り立つ． □

[系2] 任意の二つのロボット $r, r' \in \mathcal{C}$，任意の t, t' $(t \geqq 3, t' \geqq 3)$ について，$T_r(t) = T_{r'}(t') (\neq \infty)$ が成り立つならば，任意の $u(1 \leqq u \leqq min\{t-2, t'-2\})$ について $T_r(t-u) = T_{r'}(t'-u)$ が成り立つ． □

[補題4] アルゴリズム *A_Select* の任意の実行において，カウンタ ct の値が5になるロボットは存在する．

(証明) 背理法で証明する．ある実行において，ある絶対時刻 T においてすべてのロボットのカウンタ ct の値が4以下であり，T 以降どのロボットの ct も値が変化しないと仮定する．

正常ロボットは1回目の移動イベントは無条件に実行するので，T 以前にすべての正常ロボットは少なくとも1回は移動イベントを実行している．正常ロボット r が最後に移動イベントを実行した絶対時刻を T_r とし，$T_{min} = min\{T_r | r \in \mathcal{C}\}$, $T_{max} = max\{T_r | r \in \mathcal{C}\}$ とする．$T_p = T_{min}$ を満たすロボットを p とする．$ct_q \geqq 5$ なるロボット q は存在しないので，p は直進ロボットではない．従って，p はアルゴリズムを終了していない．任意の正常ロボット r について，$r(0), r(1), r(2), r(3), r(4)$ の位置は異なること，および，T_{min} 以降に少なくとも1回は移動イベントを実行することから，p は T_{max} より後にイベントを実行したとき，$MR_p = \mathcal{C} \supseteq preMR_p$ が成り立ち，移動イベントを実行する．これは T_p の定義に矛盾する． □

以下では最初にカウンタ ct の値が5になるロボットを r^* (つまり，$T_{r^*}(5) = min\{T_r(5) | r \in \mathcal{C}\}$) とし，$\mathcal{C}^* = \{r \in \mathcal{C} | T_{r^*}(4) \leqq T_r(4) \leqq T_{r^*}(5)\}$ とする(図4)．以下では，$\mathcal{C}^*$ が求める正常ロボット解集合であることを示す．

[補題5] $T_r(i) \leqq T_{r^*}(5) < T_r(i+1)$(但し，$i \leqq 4$) を満たす任意の正常ロボット r について $T_r(i+1) \neq \infty$ が成り立つ．

(証明) 背理法で証明する．$T_r(i) \leqq T_{r^*}(5) < T_r(i+1)$ (但し，$i \leqq 4$)，かつ，$T_r(i+1) = \infty$ なる $r \in \mathcal{C}, i$ が存在すると仮定する．このような正常ロボットの集合を $\mathcal{C}'$ とする．正常ロボットは1回目の移動イベントは無条件に実行する．そこで，$\mathcal{C}'$ のうち最後の移動イベントを実行した絶対時刻が最小であるロボットを p とし，その時刻を $T_p(i)(1 \leqq i \leqq 4)$ とする (つまり，p は移動イベントを i 回だけ実行したとする)．$T_p(i) \leqq T_{r^*}(5)$ より p は直進ロボットではない．p の定義より $T_p(i)$ 以降にすべての正常ロボットは移動イベントを少なくとも1回実行する．従って $T_p(i)$ より後に，いつかは $|MR_p| \geq n-k$ かつ $MR_p \supseteq preMR_p$ が成り立ち，p は $i+1$ 回目の移動イベントを実行する．これは $T_p(i+1) = \infty$ に矛盾する． □

アルゴリズム *A_Select* では，各ロボット r が他のロボット p について，p が左折ロボットであるか，ま

た，$ct_p = 5$ であるかどうかを判定している．これは実際には次のように行う．

（i） r がこれまでに知っている p の位置のうち一直線上にない3点があれば，r は p を左折ロボットと判断する．

（ii） r がこれまでに知っている p の位置のうち $\angle p_1p_2p_3 < \pi/2$（鋭角）なる3点 p_1,p_2,p_3（但し，p の位置履歴の順に p_1,p_2,p_3 とする）があれば，r は $ct_p = 5$ と判断する．

[補題6] 任意のロボット $r \in \mathcal{C}^*$ は $ct_r = 5$ となり，アルゴリズムの実行を終了する．このとき求めた左折ロボットの集合を L_r とすると，$\mathcal{C}^* \subseteqq L_r$ が成り立つ．

(証明) 補題5より，任意の $r \in \mathcal{C}^*$ は $ct_r = 5$ となり，アルゴリズムの実行を終了する．

任意の二つのロボットを $r, r' \in \mathcal{C}^*$ とする．一般性を失うことなく $T_r(5) \leqq T_{r'}(5)$ と仮定できる．このとき，(a) $T_r(5) = T_{r'}(5)$, (b) $T_{r'}(4) < T_r(5) < T_{r'}(5)$, (c) $T_r(5) = T_{r'}(4)$ のいずれかが成り立つ．

(a) 系2より，各 $i(2 \leqq i \leqq 5)$ について $T_r(i) = T_{r'}(i)$ である．このとき，r' は $\{r(j)|1 \leqq j \leqq 5\}$ を知る．また，r は $\{r'(j)|1 \leqq j \leqq 5\}$ を知る．

(b−1) ある $t(2 \leqq t \leqq 4)$ が存在し，各 $i(2 \leqq i \leqq t)$ について $T_r(i) = T_{r'}(i)$，各 $i(t+1 \leqq i \leqq 5)$ について $T_r(i) < T_{r'}(i)$ を満たすとき，r' は $\{r(j)|1 \leqq j \leqq 5, j \neq t\}$ を知る．また，r は $\{r'(j)|1 \leqq j \leqq 4\}$ を知る．

(b−2) 各 $i(3 \leqq i \leqq 5)$ について $T_{r'}(i-1) < T_r(i) < T_{r'}(i)$，かつ，$T_r(2) < T_{r'}(2)$ のとき，r' は $\{r(j)|2 \leqq j \leqq 5\}$ を知る．また，r は $\{r'(j)|2 \leqq j \leqq 4\}$ を知る．

(b−3) ある $t(2 \leqq t \leqq 3)$ が存在し，各 $i(2 \leqq i \leqq t)$ について $T_r(i+1) = T_{r'}(i)$，各 $i(t+1 \leqq i \leqq 4)$ について $T_{r'}(i) < T_r(i+1) < T_{r'}(i+1)$ を満たすとき，r' は $\{r(j)|2 \leqq j \leqq 5\}$ を知る．また，r は $\{r'(j)|1 \leqq j \leqq 4, j \neq t\}$ を知る．

(c) 系2より，各 $i(2 \leqq i \leqq 4)$ について $T_r(i+1) = T_{r'}(i)$ である．このとき，r' は $\{r(j)|2 \leqq j \leqq 5\}$ を知る．また，r は $\{r'(j)|1 \leqq j \leqq 4\}$，または，$\{r'(j)|1 \leqq j \leqq 5, j \neq 4\}$ を知る．

以上から，r は少なくとも三つの一直線上にない r' の位置を知ることができ，r' も少なくとも三つの一直線上にない r 位置を知ることができる．従って，r,r' は互いに他が左折ロボットであると判断する．すなわち $\mathcal{C}^* \subseteqq L_r$ が成り立つ． □

[補題7] $T_{r^*}(4) \leqq T_r(3) \leqq T_{r^*}(5) < T_r(4)$ を満たす任意のロボット r は直進ロボットである（図4）．

(証明) (a) $T_{r^*}(4) = T_r(3)$, (b) $T_{r^*}(4) < T_r(3) < T_{r^*}(5)$, (c) $T_r(3) = T_{r^*}(5)$ のいずれかが成り立つ．また，補題5より $T_r(4) \neq \infty$ である．

(a) 系2より $T_{r^*}(3) = T_r(2)$ が成り立つ．r は $T_r(4)$ までに，$\{r^*(j)|2 \leqq j \leqq 5, j \neq 4\}$ を知る．

(b) 系1より $T_{r^*}(3) \leqq T_r(2)$ が成り立つ．

(b−1) $T_{r^*}(3) = T_r(2)$ であるとき，r は $T_r(4)$ までに $\{r^*(j)|2 \leqq j \leqq 5, j \neq 3\}$ を知る．

(b−2) $T_{r^*}(3) < T_r(2)$ であるとき，r は $T_r(4)$ までに $\{r^*(j)|3 \leqq j \leqq 5\}$ を知る．

(c) 系2より $T_r(2) = T_{r^*}(4)$ が成り立つ．r は $T_r(4)$ までに $\{r^*(j)|3 \leqq j \leqq 5\}$ を知る．

以上から，r は $T_r(4)$ までに $\{r^*(j)|2 \leqq j \leqq 4\}$ のうち少なくとも二つの位置と $r^*(5)$ を知る．従って，r は $ct_{r^*} = 5$ であると判断し，$T_r(4)$ において直進する． □

また補題3より次の補題が明らかに成り立つ．

[補題8] $T_{r^*}(5) < T_r(3)$ を満たす任意のロボット r について $T_r(4) = \infty$ である（図4）． □

[補題9] $|\mathcal{C}^*| \geqq n - k$ が成り立つ．

(証明) $T_{r^*}(4) \leqq T_r(4) < T_{r^*}(5)$ を満たすロボット r が $n-k$ 台以上存在することを示す．

性質1より，$[T_{r^*}(1), T_{r^*}(2))$ に少なくとも $n-k$ 台以上のロボットが移動イベントを実行している．性質2より，これらのロボットは $[T_{r^*}(2), T_{r^*}(3)), [T_{r^*}(3), T_{r^*}(4)), [T_{r^*}(4), T_{r^*}(5))$ にそれぞれ1回移動イベントを実行する．よって，r^* が最初に $ct_{r^*} = 5$ となることから，$T_{r^*}(4) \leqq T_r(4) < T_{r^*}(5)$ となる．r は $n-k$ 台以上存在する． □

[定理1] $0 \leqq k < \lceil n/2 \rceil$ であるとき，アルゴリズム *A_Select* は k 個の故障にかかわらず正常ロボット選出問題を解く．

(証明) 補題7,補題8から $\mathcal{C} \backslash \mathcal{C}^*$ の任意のロボットは左折ロボットではない．よって，補題6により任意のロボット $r \in \mathcal{C}^*$ は $L_r = \mathcal{C}^*$ を解集合としてアルゴリズムを終了する．補題9より $|L_r| \geqq n-k$ である．補題7より $T_{r^*}(4) \leqq T_r(3) \leqq T_{r^*}(5) < T_r(4)$ を満たす任意のロボット r は $T_r(4)$ において，自分が解集合に含まれないことを知り，アルゴリズムを終了する．また補題8より $T_{r^*}(5) < T_r(3)$ を満たす任意のロボット r は最終状態に遷移することはない． □

また，次の定理が明らかに成り立つ．

[定理2] アルゴリズム *A_Select* の任意の実行において，どのロボットもたかだか5回しか移動イベントを実行しない． □

4. 故障耐性のためのアルゴリズム変換

4.1 アルゴリズム変換

故障ロボットが存在しない状況で問題 P を解くアルゴリズムを A とする．ここでは，アルゴリズム A を故障にかかわらず問題 P を解くアルゴリズム A^* に変換する手法について述べる．A^* では，まずアルゴリズム *A_Select* を実行して正常ロボット集合 $\mathcal{M}$ を選出し，$\mathcal{M}$ に A を実行させる．$\mathcal{M}$ による A の実行のみに着目したとき，これが実行に関する条件 Ψ^P を満たしているならば，A^* は問題 P を解くとみなす．

簡単のため，アルゴリズム A での各ロボット r のプログラムは図5の形式で与えられると仮定する．つまり，ロボットはまず起動時に移動し，それ以降の移動はすべてのロボットの移動を認識してから移動する．但し，図5において r が使用する変数 $flag_A$, MR_r はアルゴリズム *A_Select* と同様に使用される．また，$Action_A$ では r の内部状態（$\mathcal{N}$ の各ロボットに関するそれ以前に視覚センサから得た位置履歴を含む）から次の動作を決定し，実行する．つまり，具体的に $Action_A$ を決めることによってアルゴリズムが決まる．

[アルゴリズム A^* （各ロボット r の動作）]

r が使用する各変数はアルゴリズム *A_Select* と同様．

(a) $ct_r \leqq 4$ のとき：*A_Select* と同様に動作する．

(b) $ct_r = 5$ のとき：左折ロボットの集合 L_r を求めていなければ，L_r を求める．また，$MR_r \supseteqq L_r$，かつ，任意のロボット $q \in L_r$ について $ct_q \geqq 5$（r は q の位置履歴から，$ct_q \geqq 5$ かどうかを判定できる（後述の仮定 (A)））が成り立つまで，視覚センサによる観測を繰り返す．$MR_r \supseteqq L_r$，かつ，任意のロボット $q \in L_r$ について $ct_q \geqq 5$ が成り立てば，L_r に対してアルゴリズム A の実行を開始する．つまり，r の内部状態をアルゴリズム A の初期状態に設定し(注4)，以降は L_r 以外のロボットを無視して A を実行する． □

アルゴリズム A^* （r のプログラム）の詳細を図6に示す．$Wait_{A\_Select}$, $Action_{A\_Select}$, $Wait_A$ はそれぞれ図2，図5に示した手続きである．

カウンタ ct の値が4以下のときの A^* の動作は *A_Select* と同じである．補題4よりカウンタ ct の値が5になるロボットは存在する．最初に $ct = 5$ になるロボットを r^* で表し，$\mathcal{C}^* = \{r \in \mathcal{C} | T_{r^*}(4) \leqq T_r(4) \leqq T_{r^*}(5)\}$ とする．$\mathcal{C}^*$ は *A_Select* が求める正常ロボット解集合である．

$\mathcal{C}^*$ に属する任意のロボットを r, r' とする．アルゴリズム A^* では $\mathcal{C}^*$ を利用してアルゴリズム A を実行するが，$T_r(5) = T_{r'}(5) < T_r(6) < T_{r'}(6) \neq \infty$ のとき，r' は r の位置 $r(5)$ を知ることができないことがある．このため，$r(4) = r(6)$ が成り立つと r' は r の移動を検知できず，A^* はデッドロック状態に陥って

```
A() {
  watch;
  Action_A(W);
  do {
    Wait_A(W);
    Action_A(W);
  } while (flag_A);
}
```

(a) Main Program

```
Wait_A(G) {
  MR_r := {r};
  do {
    watch;
    update MR_r;
  } while (MR_r ⊂ G);
}
```

(b) Procedure $Wait_A$

図5 アルゴリズム A(ロボット r)
Fig. 5 Algorithm A (for robot r).

```
A*() {
  watch;
  Action_A_Select();
  do {
    Wait_A_Select();
    Action_A_Select();
  } while (flag_A_Select);
  if (ct_r=5) then {
    while ¬(∀q∈L_r[ct_q≧5]) do
      watch;
    set r's state to an initial state of A;
    Action_A(L_r);
    do {
      Wait_A(L_r);
      Action_A(L_r);
    } while (flag_A);
  }
}
```

図6 アルゴリズム A^*(ロボット r)
Fig. 6 Algorithm A^* (for robot r).

(注4)：直前の *watch* は A の最初の *watch* に相当するので，そこで得られた L_r のロボットの位置情報は初期化してはいけない．

しまう．また，A^* では，どのロボットが移動したかを知る必要がある．すなわち，A^* は追跡可能である必要がある．

そこで次の仮定をおく．

(A) $T_r(6) \neq \infty$ なる任意のロボット r について，$r(4) \neq r(6)$ が成り立つ．

(B) アルゴリズム A^* は追跡可能である．

故障ロボットが存在しない状況で問題 P を解くアルゴリズムが，仮定(A),(B) を満たさなくても，アルゴリズム *A_Select* のようにロボットの移動を制限することにより，仮定 (A),(B) を満たすように変形できることが多い．例えば，文献[4],[5]の各ロボットの原点や単位距離を一致させるアルゴリズムは，仮定(A),(B) を満たすように容易に変形できる．

以下では，上記の変換で得られた A^* が故障にかかわらず問題 P を解くことを述べる．先に述べたように，A^* で選出された正常ロボット集合 $\mathcal{M}$ の実行が，実行に関する条件 Ψ^P を満たしているならば，A^* は問題 P を解くとみなす．つまり，次に定義する射影実行という概念に関して，定理3が成り立つ．

[定義7] (射影実行) ロボット集合を $\mathcal{N} = \{r_1, r_2, \cdots, r_n\}$，アルゴリズムの実行を $E = \gamma_0, \gamma_1, \gamma_2, \cdots$，$\mathcal{N}$ の空でない部分集合を $\mathcal{H} = \{r_{i_1}, r_{i_2}, \cdots, r_{i_h}\}$ ($\mathcal{H} \subseteq \mathcal{N}, h \neq 0, i_1 < i_2 < \cdots < i_h$) とする．

E の大域状況 γ_j から，$\mathcal{H}$ に属さないロボットの状態と位置を取り除いたものを γ'_j とする．すなわち，$\gamma_j = \langle \alpha_1, \beta_1, \alpha_2, \beta_2, \cdots, \alpha_n, \beta_n \rangle$ とするとき，$\gamma'_j = \langle \alpha_{i_1}, \beta_{i_1}, \alpha_{i_2}, \beta_{i_2}, \cdots, \alpha_{i_h}, \beta_{i_h} \rangle$ である．γ'_j は $\mathcal{H}$ に属するロボットの状態と位置を表しているので，これを部分状況と呼ぶ．また，$E' = \gamma'_0, \gamma'_1, \gamma'_2, \cdots$ とする．

このとき，$E' = \gamma'_0, \gamma'_1, \gamma'_2, \cdots$ に同じ部分状況が連続して現れる場合，これらを一つの部分状況に置換して得られる部分状況の系列を E の $\mathcal{H}$ への射影実行と言い，$E|_{\mathcal{H}}$ と表す．つまり，$E|_{\mathcal{H}} = \gamma'_{j_0}, \gamma'_{j_1}, \gamma'_{j_2}, \cdots$ は次の四つの条件を満たす．

(1) $j_0 = 0$(つまり,$\gamma'_{j_0} = \gamma'_0$).

(2) 任意の $a(\geq 0)$ について $j_a < j_{a+1}$.

(3) 任意の $a(\geq 0)$ について $\gamma'_{j_a} \neq \gamma'_{j_{a+1}}$.

(4) 任意の $b(\geq 1)$ について $\gamma'_b \neq \gamma'_{b-1}$ ならば，$j_c = b$ なる c が存在する． □

[定理3] 故障ロボットが存在しない状況で問題 P を解くアルゴリズムを A とする．但し A は，図5の形式で与えられる．故障ロボット数がたかだか $k(0 \leq k < \lceil n/2 \rceil)$ のとき，アルゴリズム A^* の任意の実行 E において，ある正常ロボットの集合 $\mathcal{M}$ ($\subseteq \mathcal{C}, |\mathcal{M}| \geq n-k$) が存在し，$E$ の $\mathcal{M}$ への射影実行 $E|_{\mathcal{M}}$ は条件 Ψ^P を満たす． □

(注1) その解がロボットの初期位置に依存する問題も考えられる．そのような問題に対しては，任意の $r \in \mathcal{M}$ は A の実行を開始する前に，その初期位置 $r(0)$ に移動するように変更することで対応できる．

(注2) A^* では，正常ロボット解集合 $\mathcal{M}$ に属さない正常ロボット q が左折ロボット集合 $L_q (\supseteq \mathcal{M} \cup \{q\})$ を求めてしまうことがある．これは，例えば $\mathcal{M}$ が求まった後に q が始動し，かつ，$\mathcal{M}$ に属するロボットのアルゴリズム A による動作が *A_Select* による動作と偶然に一致するときに起きる．このとき，ロボット q は，*A_Select* 実行後に，L_q に対してアルゴリズム A を実行する．アルゴリズム A では，移動後にすべてのロボットの移動を認識してから次の移動を行うので，アルゴリズム A の実行を遅れて開始したロボット q の A に関する移動回数は，正常ロボット解集合 $\mathcal{M}$ に属するロボットの A に関する移動回数よりも少ない．従って，最終状態に達する（つまり，解を求めて停止する）までのロボットの移動回数が，L_q に対して A を実行した場合の方が，$\mathcal{M}$ に対して A を実行した場合より少なくなければ(注5)，このようなロボット q は最終状態に達しない．

4.2 変換の適用例：合意問題

[定義8] (合意問題) 合意問題とは，各ロボット $r \in \mathcal{N}$ が初期データ $I_r (\in \{0,1\})$ をもつとき，すべてのロボットが同じ値 $O(\in \{0,1\})$ を決定して停止する問題である．但し，すべてのロボットの入力データが同じ(その値を I とする)なら，$O = I$ でなければならない． □

各ロボット r は任意の方向に進んだ後，$I_r = 0$ なら左折，$I_r = 1$ なら右折することにより，自分の初期データ I_r を他のロボットに知らせることができる．従って，故障ロボットが存在しなければ，合意問題は容易に解ける．合意問題を解くためのアルゴリズム *Agreemet* は，A の手続き $Action_A$ を図7に示す手続き $Action_{Agreement}$ に置換したものである．

アルゴリズム *Agreement* は左折／右折によって情報を伝えているので，移動距離を制限することにより，仮定 (A), (B) を満たすように変更できる．従って，上で述べた変換で得られるアルゴリズム *Agreement**

(注5)：移動回数がロボットの総数に対して単調非減少であるときなど．

```
Action_Agreement(G) {
  switch (ct_r) {
    case 0 :
      d_r:=(distance between r and the nearest
            robot on P_r(1))/16;
      move to (d_r,0);   ct_r:=1;
    case 1 :
      move to (2d_r,0);  ct_r:=2;
    case 2 :
      if (I_r=0) then
        move to (2d_r,d_r);
      else
        move to (2d_r,-d_r);
      ct_r:=3;
    case 3 :
      if (The number of left-turned robot
          >|G|/2) then O:=0;
      else O:=1;
      flag_A:=false;
  }
}
```

図7 手続き $Action_{Agreement}$(ロボットr)
Fig. 7 Procedure $Action_{Agreement}$(for robot r).

は故障にかかわらず合意問題を解くアルゴリズムである.

5. むすび

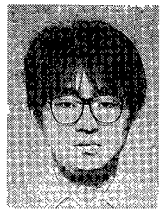
本論文では,初期停止故障ロボットを含むロボット群で,協調して問題を解くためのアルゴリズムについて考察した.特に,故障耐性のないアルゴリズムを故障耐性のあるアルゴリズムに変換する方法について検討を加えた.今後の課題として,初期停止故障以外の故障を考慮することなどが考えられる.

謝辞 日ごろより有益な御討論を頂きました奈良先端大・情報科学研究科の藤原暁宏君に感謝致します.なお,本研究の一部は,平成6年度文部省科学研究費補助金(奨励(A)06780262)の援助を受けている.

文献

[1] H. Ando, I. Suzuki, and M. Yamashita, "Formation and agreement problems for synchronous mobile robots with limited visibility," IEEE Trans. on Robotics and Automation (submitted).
[2] M.J.Fischer, N.A.Lynch, and M.S.Paterson, "Impossibility of distributed consensus with one faulty process," JACM, vol.32, no.2, pp.374–382, April 1985.
[3] K. Sugihara and I. Suzuki, "Distributed motion coordination of multiple mobile robots," Proc. of the 5th IEEE International Symposium on Intelligent Control, pp.138–143, Sept. 1990.
[4] I. Suzuki and M. Yamashita, "Cooperative control algorithms for anonymous mobile robots," 第6回回路とシステム軽井沢ワークショップ論文集, pp.523–530, 1993.
[5] I. Suzuki and M. Yamashita, "Formation and agreement problems for anonymous mobile robots," Proc. of 31st Annual Allerton Conf. on Communication, Control and Computing, pp.93–102, Oct. 1993.
[6] 吉田大輔, "自律移動ロボット群を円状配置する協調分散型解法," 大阪大学基礎工学部特別研究報告, March 1993.
[7] 吉田大輔, "停止故障耐性を考慮した自律移動ロボット群のための協調問題解法について," 情報処理学会研究報告, 94-AL-42, pp.63–70, Nov. 1994.

(平成7年3月1日受付, 12月4日再受付)



吉田 大輔

平5阪大・基礎工・情報卒.平7奈良先端大博士前期課程了.同年(株)日立製作所入社.現在,仮想計算機の開発に従事.情報処理学会会員.

増澤 利光 (正員)

昭57阪大・基礎工・情報卒.昭62同大大学院博士課程了.工博.同年同大情報処理教育センター助手.同大基礎工学部助教授を経て,平6奈良先端大・情報科学助教授,現在に至る.平5コーネル大学客員準教授(文部省在外研究員).分散アルゴリズム,並列アルゴリズムに関する研究に従事.ACM, IEEE, EATCS, 情報処理学会各会員.

藤原 秀雄 (正員)

昭44阪大・工・電子卒.昭49同大大学院博士課程了.同大・工・電子助手,明治大・工・電子通信助教授,情報科学教授を経て,現在奈良先端大・情報科学教授.昭56ウォータールー大客員助教授.昭59マッギル大客員準教授.論理設計論,フォールトトレランス,設計自動化,テスト容易化設計,テスト生成,並列処理,計算複雑度に関する研究に従事.著書「Logic Testing and Design for Testability」(MIT Press)など.大川出版賞.情報処理学会会員.IEEE Fellow.