エリアカバレッジタスクにおける回転動作を考慮した移動ロボットの動作計画決定手法

矢野 泰生[1,a)]　高瀬 英希[1]　高木 一義[1]　高木 直史[1]

概要：近年，室内清掃のようなエリアカバレッジタスクにおけるモバイルロボットの活用が検討されてきている．単体ロボットによるエリアカバレッジの場合，作業空間を格子グラフとして考え，そのグラフ上で全域木を構成することで作業空間全体を網羅する巡回経路を得られることが知られている．この巡回経路は経路長という観点では重複なしの最短経路となっているが，必ずしも時間効率の観点で最適とは限らない．そこで本研究では，経路上での回転動作に着目して格子グラフ上の全域木を探索することで時間的に最短な経路を導出する．具体的には，回転動作を考慮したグラフの簡略表現を用いることで最適な木を高速に探索できる手法を提案する．また単体ロボットを対象として得られる最適解から複数ロボットの場合の動作計画を決定する手法についても検討する．

キーワード：移動ロボット， 経路計画， 全域木

A motion planning method for mobile robot considering rotational motion in area coverage task

Yano Taiki[1,a)]　Takase Hideki[1]　Takagi Kazuyoshi[1]　Takagi Naofumi[1]

Abstract: In recent years, utilization of mobile robot in area coverage tasks has been studied. In the case of area coverage by a single robot, it is possible to obtain a traveling route for covering the entire work space by constructing a spanning tree on a grid graph representing the work space. The tour route obtained by the above method is the shortest one in terms of the route length. However, it is not optimal in terms of time efficiency when we consider the rotation behavior of the robot. In this research, we focus on the rotational motion on the route. We propose a method for constructing the spanning trees on the lattice graph to derive the optimal time efficient path. Proposed method further searches the optimal tree using the simplified representation of the graph considering rotational motion. We also discuss the method of determining the motion plan for multiple robots using the optimal tree for a single robot.

Keywords: mobile robot, path planning, spanning tree

1. はじめに

近年，高い機動性を有するモバイルロボットの活用が検討されてきている．モバイルロボットの活用が期待される例としては，オフィスフロアや工場における室内清掃作業，施設の巡回警備やメンテナンスおよび大規模農場における作物育成補助などが挙げられる．このような活用例では，ロボットは指定された範囲内の全領域を網羅するように移動しながら作業を行うことが求められる．このような作業はエリアカバレッジタスクと呼ばれる．また，エリアカバレッジタスクにおけるロボットの移動経路を求める問題は，ロボットによるエリアカバレッジ問題と呼ばれる．

ロボットによるエリアカバレッジ問題において，領域の網羅率および経路の重複の少なさに着目した経路長最短経路を導出するため，これまでに様々な研究が行われている．特に，単体ロボットによるエリアカバレッジ問題においては，対象の作業空間を格子グラフとして表現し，そのグラフ上の全域木を考えることで経路長最短の経路を得られることが知られている [1]．この経路は，グラフで表現される作業空間を重複無しで網羅することが可能である．

一方，経路上でのロボットの実際の走行時間を考える場合，即ちエリアカバレッジタスクの時間効率を考える場合

[1] 京都大学大学院情報学研究科
Graduate School of Informatics, Kyoto University
[a)] emb@lab3.kuis.kyoto-u.ac.jp

は，経路の重複を考慮するだけでは不十分である．実際には経路上での回転動作により，同じ長さの経路においてもタスク完了までに要する時間には差が出てくるためである．そのため最短時間の経路を求めるためには，経路上での回転動作を考慮した問題モデルおよび解の探索が必要となる．しかし，単純なグラフ表現を用いた問題モデルにおいては，グラフの形状にもよるが，グラフのサイズが増大するに従って最適解を得るために探索すべき解の個数が爆発的に増加するという問題が存在する．また経路上での実走行時間を正確に見積もるためにより現実に即した問題モデルを利用する手法も研究されているが，この場合は僅かな未到達領域のために大幅な経路長の増加が発生するなど網羅性および経路の重複の面で最適解を決定することが難しい [2]．そのため厳密な最適解を得るのではなく，遺伝的アルゴリズムなどを用いたヒューリスティックに解を得る手法の研究も行われてきている [3]-[6]．また，時間効率を向上させるために複数のロボットを利用する手法も研究されてきている．複数ロボットの場合は作業空間の最適な分割および分割された空間内での最適な経路を決定する手法が必要となる．

本研究の目的は，移動ロボットのエリアカバレッジタスクにおいて，領域の網羅性を保ちつつ時間効率の面で最適な移動経路を決定することである．目的達成のため，まず単体ロボットについて 2 次元平面上で最適な移動経路を高速に探索するアルゴリズムを提案する．加えて，複数ロボットによるエリアカバレッジタスクについても，単体ロボットを対象として得られた最適解からそれぞれの動作計画を決定する手法について検討する．

単体ロボットについて提案するアルゴリズムでは，領域全体を網羅する経路を得るために作業空間を表す格子グラフ上での全域木の探索を行う．この探索においては，時間効率最適の経路を得るために経路上での回転動作の回数に着目した探索を行う．単体ロボットの場合は，最短の経路長を保持しつつ経路上での回転動作の回数を最小化することで時間的に最短な経路が得られる．また格子グラフ上の全域木に対して，必要な性質を損なわないように辺の追加または除去を行った全域木の簡略表現を利用する．この簡略表現は，回転の回数が等しい複数の全域木を表現することが可能であり，探索のための計算量を大幅に削減できる．そのため，ある程度大きいサイズの問題に対しても現実的な計算時間で最適解を求めることが可能となる．

本稿の構成は以下の通りである．まず 2 章で既存研究を紹介し，その問題点について述べる．次に 3 章において，本研究で取り扱うエリアカバレッジ問題を数理的にモデル化し，その特性について考察する．4 章では 3 章で与えたモデルにおける全域木の探索アルゴリズムを提案し，アルゴリズムの計算量および正当性についても議論する．さらに 5 章では，単体ロボットを対象として提案するアルゴリズムを適用して得られた最適解から，複数ロボットの動作計画を決定する手法について検討する．6 章では結論として全体のまとめと今後の課題について述べる．

2. 関連研究

ロボットによるエリアカバレッジタスクに関する既存研究を紹介する．初期の研究においては対象エリア内でロボットをランダムに移動させる手法が検討されている [7]．この手法は未到達領域について考慮しないため，経路に重複が生じており非効率的である．経路上での重複を考慮しつつ，単体ロボットを対象とした作業空間を網羅する経路を得る手法として文献 [1], [8] の手法が存在する．これらの手法では作業空間を格子グラフとしてモデル化して経路を求めている．特に文献 [1] の手法, Spanning Tree Coverage（STC）は格子グラフ上の全域木を考えることで作業空間を網羅する経路長最短の経路を得ることが可能である．しかし，この手法はロボットの経路上での回転動作を考慮していないため，時間効率の面で最適な経路が得られるとは限らない．経路上の回転動作を最小化することを目的とした手法として文献 [2] の手法が存在する．この手法では，格子グラフよりも現実に即した問題モデルを利用しており網羅性と経路長を両立した最適解を得ることが難しい．

STC を応用して複数ロボットの経路を求める手法として文献 [9], [10] の手法が存在する．文献 [10] では複数ロボットによるエリアカバレッジが NP 完全であることが示されている．これらの手法はヒューリスティックな手法となっており，必ずしも最適解を得られるとは限らない．

また，単体ロボットおよび複数ロボットの場合に，遺伝的アルゴリズムを利用してヒューリスティックに解を探索する試みも行われている [3]-[6]．

以上で述べてきた既存研究によって，単体ロボットによるエリアカバレッジ問題に対して，エリア内の全ての領域を通過する完全カバレッジ経路を導出する手法が与えられている．またこの時，経路長最短の経路を発見する手法が提案されている．時間効率や電力効率の面では，最適解を保証する手法ではなく，より良い解を探索するヒューリスティックな手法が提案されている．複数台ロボットの場合においては，STC の応用によって作業空間を網羅した経路を導出することが可能である．また，複数ロボットの場合のエリアカバレッジ問題が NP 完全であることが示されており，遺伝的アルゴリズムなどを利用した手法が提案されてきている．しかし，これらの手法では探索に要する時間が現実的でなくなる可能性や解が局所最適に陥る可能性がある．

3. 回転動作を考慮したエリアカバレッジ問題

本章では，単体ロボットによるエリアカバレッジ問題について定義する．その後，定義した問題の数理的なモデル

を与え，その性質について説明する．

3.1 エリアカバレッジタスク

まず，エリアカバレッジタスクの対象空間について以下の 5 つの仮定を与える．

- 対象空間は大きさの等しい正方形セルに分割できる
- 正方形セル同士は格子状に整列される
- 対象空間は連結であり，任意の 2 つの地点を結ぶ経路が必ず存在する
- 正方形セルは 4 つの正方形かつ大きさの等しいサブセルに分割できる
- サブセルの大きさはロボットが一度にカバー可能な大きさである

これは対象空間がグラフで表現可能であることを表している．また各正方形セルの中心を頂点として考えた全域木から，経路長最短の空間全体を通過する経路を得られることを表している．**図 1** に対象空間の例を示す．

次に，本研究で対象とするロボットの動作について説明する．ロボットが行う動作は，直進移動および移動を伴わない左右への 90 度回転のみとする．対象のロボットは一般的な家庭用清掃ロボットを想定することとすれば，隣接するサブセル間の直進を直進移動の 1 単位として考えた時，経路上での直進移動および回転移動に要する時間の間に大きな偏りは存在しないものと考えられる．そのため経路上での走行時間を考える場合は，経路上での直進移動の回数および回転動作の回数の両方が重要な要素となってくると考えられる．

これらの仮定の下で，本研究で取り扱うエリアカバレッジ問題は，エリアカバレッジタスクの対象エリアが与えられた時に，解として全サブセルを被覆するロボットの巡回経路を求める問題として定義される．

本問題における最適解は最小のカバレッジ時間を与えるロボットの巡回経路である．ここでカバレッジ時間とは，エリアカバレッジタスクの完了までに要する時間を表している．ロボットが巡回経路を 1 周するのに要する時間は，経路上での直進動作の回数および経路上で行った回転動作の回数によって定まる．

3.2 問題の定式化

3.1 節で述べた単体ロボットによるエリアカバレッジ問題を，与えられた格子グラフから全域木を得るグラフの問題としてモデル化する．まず利用する定数および変数の記号の定義を**表 1** のように与える．定義した記号を利用して問題のモデルを以下のように与える

- インスタンス: 格子グラフ $G = (V,\ E)$
- 解　　: 全域木 $T = (V,\ E_t)$
- 最適化目標　: $\min\{Cost(T)\}$

以下にインスタンス，解および最適化目標のそれぞれについて詳述する．

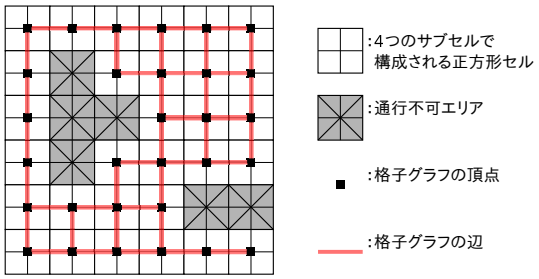

図 1　対象空間と格子グラフの例.
Fig. 1　Example of the problem.

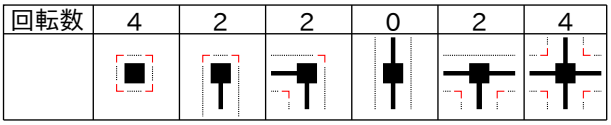

図 2　90 度回転の回数による頂点の分類.
Fig. 2　Classification of vertices on lattice.

問題のインスタンスは，図 1 に赤線および黒い正方形で示しているように対象エリアの地形データを反映した格子グラフ G である．格子グラフの頂点は対象エリア上の正方形セルに対応している．辺は正方形セル同士の接続状況を表しており，通行可能な正方形セル間にのみ辺が存在する．

本問題の解はグラフ上の全域木である．2 章で述べたように，全域木からはその木の外周に沿ったロボットの巡回経路を導出することができる．そのためグラフ上で全域木を得ることができれば，対象エリアの全体をカバーする巡回経路を導出することが可能である．

本問題における最適化目標は，解として得られた全域木 T のコスト $Cost(T)$ の最小化である．木のコストは頂点数に比例するコスト $4C_S\|V\|$ および各頂点の持つ回転の回数の総和に比例するコスト $C_R R(t)$ という 2 つのコストの和として求められる．前者のコストは巡回経路の長さによるコストである．全域木の 1 つの頂点はロボットの経路上では 4 つのサブセルに対応しているため，経路長に対応する値として頂点数の 4 倍が与えられている．後者のコスト $C_R R(t)$ は，経路上でロボットが行う回転の回数に対応し

表 1　定数および変数記号の定義.
Table 1　Definition of constants and variable symbols.

記号	定義
V	頂点集合.
E	辺集合.
E_t	辺集合の部分集合．$E_t \subseteq E$
G	入力グラフ．$G = (V,\ E)$
T	木．$T = (V,\ E_t)$
	$Cost(T) = 4S_c\|V\| + R_c R_t$
C_S	ロボットの直進移動動作に要するコストの単位量.
C_R	ロボットの 90 度回転動作に要するコスト.
$R(T)$	木 T に含まれる頂点が持つ 90 度回転の回数の総和.

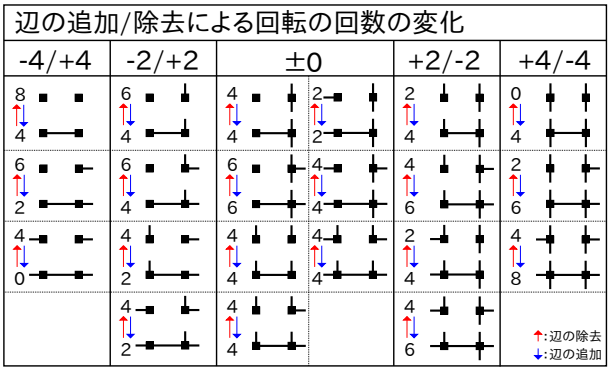

図 3 追加/除去コストによる辺の分類.
Fig. 3 Classification of edges on lattice.

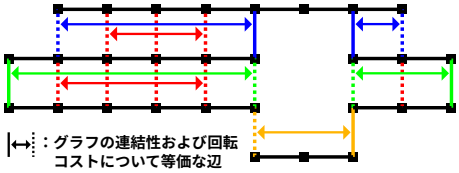

図 4 コストおよび接続性の等価な辺の選択肢.
Fig. 4 Choices of edges with same cost and connectivity.

たコストである．回転の回数はロボットの 90 度回転を 1 単位とする．頂点の次数と辺の接続パターンにより各頂点の周りでの回転の回数が定まり，全域木全体の回転に起因するコストを得ることができる．**図 2** に格子グラフ上の各頂点に置ける回転の回数を示す．点線が頂点の周囲でのロボットの経路，赤色で示している部分が 90 度回転である．

3.3 問題の性質

3.2 節で説明した問題の解の個数について考察する．まず入力グラフとして $n*n$ のグリッドマップに対応したグラフを考える．$n*n$ のグリッドマップに対応したグラフの中で，グラフ上の全域木の個数が最も多くなるのは $n*n$ の正方格子グラフの場合である．文献 [11] により，頂点数 $n*n$ の格子グラフ上の全域木の個数は，有限の非ゼロの定数 Z_L によって $\exp(n^2 Z_L)$ と近似することができる．従って，対象空間のサイズの増加に対して全域木の個数は指数関数的に増加することとなる．

次に，格子グラフ上の辺の種類について考える．格子グラフ上の辺は，両端の頂点の種類によって，その辺を追加または除去した場合の回転コストの変化量が異なる．辺を追加または除去した場合の回転の回数の変化量を辺の追加コストまたは辺の除去コストと呼ぶこととする．追加コストおよび除去コストによる辺の分類を**図 3** に示す．図中で，各辺は辺が存在しない場合および存在する場合の上下 2 段で表現されており，その横に辺の追加または除去による回転の回数の変化を示している．

ここで全域木に含まれる辺について，経路上での回転の回数およびグラフの接続性に着目して考える．**図 4** に，グラフの連結性および回転コストについて等価な辺の組み合わせを示す．実線が存在している辺，同じ色の点線が辺の有無を入れ替え可能な辺の位置の選択肢である．これらの辺の組み合わせは，いずれの辺が木に存在するかを選択することにより，複数の同じ回転コストを持つ木を構成することが可能である．従ってこれらの辺の組み合わせは，回転コスト最小の全域木を探索する上で探索量を不必要に増加させる要因になっていると考えられる．

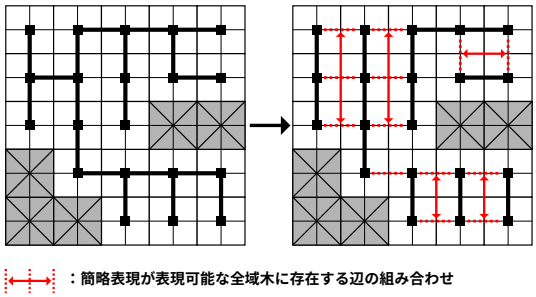

図 5 全域木の簡略表現.
Fig. 5 Simplified representation of a spanning tree.

また，図 4 で赤色または青色で図示されている辺の組み合わせは，それぞれ $+4/-4$ の辺の組み合わせおよび $+2/-2$ の辺の組み合わせである．これらは除去コストがマイナスの辺であるため，回転コスト最小の全域木の探索においては可能であれば存在しないほうが望ましいものである．グラフの連結性のためにいずれかの辺が存在する必要がある場合は，探索量を増加させないために，いずれの辺が存在するのかを選択することなく，辺の組み合わせ全体を 1 つのまとまりとして考えることで探索量を削減できると考えられる．これは 0/0 の辺の組み合わせにおいても同様である．

4. 単体ロボットにおける探索アルゴリズム

本章では，単体ロボットによるエリアカバレッジ問題のための全域木の探手法を提案する．まず，3.3 節で説明した辺の追加または除去に関する特性を利用し，計算量を削減できるグラフの簡略表現を示す．続いて，簡略表現の特徴を利用した最適な木の探索手法について説明する．

4.1 グラフの簡略表現

本問題において，解の評価基準は頂点数に起因するコストおよび回転コストの和である．ここで全域木に含まれ

表 2 グラフ G' の簡略表現 F.
Table 2 The simplified representation F of a graph G'.

記号	定義
F	$F = (V,\ E_f)$
E_f	辺集合．$E_f \subseteq E$ ただし，全ての $e \in E_f$ に対して追加コストは 0 以下

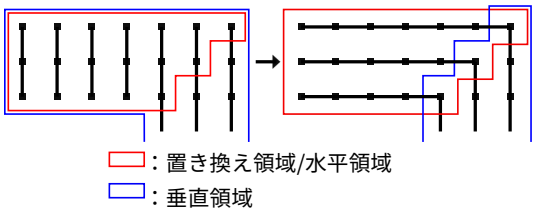

図 6 領域置き換えの例.
Fig. 6 Example of region replacement.

る頂点数は入力グラフの頂点数と等しく一定なので，解の最適化において考慮すべきパラメータは各頂点の持つ回転の回数の和である．そこで，グラフ G の部分グラフ $G' = (V, E'),\ (E' \subseteq E)$ の簡略表現 F を**表 2** のように定義する．簡略表現 F は，グラフ G' に対して，図 4 に示されている除去コストが 0 以下の辺の除去を繰り返し行うことで得られる．簡略表現上の辺は追加コストが 0 以下の辺のみとなる．また簡略表現 F の回転コストは，F に含まれる各頂点に対して，図 2 に示されている頂点の種類を考えることで決定される．簡略表現の例を**図 5** に示す．左側の全域木から図 4 に示されている除去コスト 0 以下の辺を除去していくことで，右側の簡略表現が得られる．この簡略表現は，赤色の点線および矢印で示されている辺の組み合わせにおける存在する辺の選択によって，$3*3*2*2*2 = 72$ 通りの全域木を表現している．

ここで簡略表現上で辺の接続の形による各頂点の向きを考える．次数 2 で，2 本の辺が垂直に接続されている頂点を垂直頂点，2 本の辺が水平に接続されている頂点を水平頂点とする．またこれ以外の頂点は全て垂直および水平の両方向を持つ水平垂直頂点とする．グラフ上で隣接している水平頂点および垂直頂点によってグラフ上の水平領域および垂直領域を考える．この時，水平垂直頂点は水平領域および垂直領域に重複して属する．各領域の持つ水平又は垂直方向の長さを領域の長さ，直行する方向の幅を領域の幅とする．ここで，回転コスト最小の簡略表現と現在の簡略表現との間の差を，方向の一致しない頂点の集合として考える．ある簡略表現から回転コスト最小の簡略表現を得る操作を，次節のアルゴリズムでは簡略表現上のいくつかの領域の部分領域の方向を置き換えていく操作として考えている．この置き換えられる領域を置き換え領域と呼ぶこととする．簡略表現上の回転コストは，領域の重複部分における接続辺および同方向の頂点によって構成される線分の本数によって決定される．回転コストを減少させるような置き換え領域の選択は，領域間の接続辺に注意しながら簡略表現上の線分数を減少させる様に行われる．領域置き換えの例を**図 6** に示す．領域内の頂点方向の置き換えにより，領域内に存在する辺が変化し，領域の重複部分における接続の形が変わっている．また，図 6 に表示されている部分において，回転コストに影響を与えている線分数は置き換えにより 7 本から 6 本に減少している．回転の回数は 22 回から 12 回に減少している．4.2 節で説明するアルゴリズムは，この簡略表現の特徴を利用して解の探索を行う．

アルゴリズム 1 格子グラフからの回転コスト最小となる全域木の導出.

Input: 格子グラフ $G = (V, E)$
Output: 回転コスト最小の全域木 T

```
1: L ⇐ GET_LINES(G)
2: while L ≠ null do
3:     l' ⇐ l ∈ L
4:     L ⇐ L − {l'}
5:     d ⇐ DIRECTION(l')
6:     (R, cost, reg_num) ⇐ SEARCHCHANGEREGION(l', d)
7:     if (cost < 0) or (cost = 0 and reg_num < 0) then
8:         G ⇐ CHANGEDIRECTION(G, R, d)
9:         L ⇐ GET_LINES(G)
10:    end if
11: end while
12: T ⇐ CONNECTREGS(G)
```

アルゴリズム 2 線分 l を起点とした回転コスト減少量最大の置き換え領域の導出 SearchChangeRegion.

Input: 線分 l および線分 l の方向 d
Output:
回転コスト減少量最大の置き換え領域 $region$，回転コストの減少量 $cost$ および連結なグラフの個数の減少量 reg_num.

```
1: region ⇐ l
2: tmp_reg ⇐ region
3: cost ⇐ 0
4: tmp_cost ⇐ 0
5: reg_num ⇐ 0
6: tmp_r_num ⇐ 0
7: loop
8:     next_vertices ⇐ NEXTVERTICES(tmp_reg, d)
9:     if next_vertices = null then
10:        break
11:    end if
12:    (add_vertex, add_cost, add_r_num)
                    ⇐ MINADDCOST(tmp_reg, next_vertices, l, d)
13:    tmp_cost ⇐ tmp_cost + add_cost
14:    tmp_reg ⇐ tmp_reg ∪ {add_vertex}
15:    tmp_r_num ⇐ tmp_r_num + add_r_num
16:    if (tmp_cost < cost) or
          ( tmp_cost = cost and tmp_r_num < reg_num) then
17:        cost ⇐ tmp_cost
18:        region ⇐ tmp_reg
19:        reg_num ⇐ tmp_r_num
20:    end if
21: end loop
```

4.2 全域木探索アルゴリズム

提案するアルゴリズムは，格子グラフ上の各線分に対して回転コストの減少する置き換え領域または回転コストを保存しつつ連結なグラフの個数を減少させる置き換え領域を探索し，発見した置き換え領域を順次置き換えていく処理となっている．

アルゴリズム 1 に格子グラフ G から回転コスト最小の全域木を求めるアルゴリズムを示す．アルゴリズム中で使用している記号は，L が格子グラフ上に存在する線分の集合，l' が 1 つの線分を表す頂点集合，d が方向を表す変数，R は置き換え領域を表現する頂点集合である．ここで，格子グラフ上に存在する線分とは，辺によって接続されている一直線上に並んだ頂点の集合である．提案するアルゴリズムでは，まず 1 行目で関数 GET_LINES によって格子グラフ G 上に存在する線分の集合 L を取得している．GET_LINES は単純にグラフ上を走査して存在する垂直線分および水平線分を返す関数である．次に，3-4 行目で線分集合 L から線分 l' を取り出している．5 行目では線分の方向を返す関数 DIRECTION によって l' の方向 d を得ている．6 行目ではアルゴリズム 2 に示す関数 SEARCHCHANGEREGION によって，与えられた線分 l' から d と直行する方向に 4.1 節で説明した置き換え領域を探索している．探索の詳細については後述する．

7-9 行目では，領域 R を置き換えた場合の回転コストの変化量 $cost$ が負の場合，または回転コストが保存された状態で連結なグラフ数の変化量が負の場合に，領域の置換えを実行し，グラフ G および集合 L を更新している．ここで，関数 CHANGEDIRECTION はグラフ G 上で置き換え領域 R 内の頂点の方向を d と直行する方向に置き換え，頂点の方向と辺の接続関係が一致するように辺の追加および除去を行う．この頂点方向の置き換えにおいて，領域内の線分の端点にあたる頂点は隣接領域との接続点になる可能性があるため垂直水平両方向の頂点に設定される．ただし，頂点一つからなる線分が複数本隣接して発生する場合は，それらの頂点をまとめて d 方向の線分とする．また，線分の端点以外の頂点から領域外に伸びる辺は全て除去コストが 0 以下であるためグラフから除去する．以上の操作を集合 L が空集合になるまで繰り返す．最後に，関数 CONNECTREGS によって簡略表現から全域木を得る．この段階で簡略表現 F が非連結な場合は，非連結なグラフ間に存在する辺の追加コストは必ず 0 より大きい．よって，連結なグラフの個数が減少しなくなるまで追加コスト +2 の辺を追加する．この時点で，まだ非連結なグラフが存在する場合は，連結なグラフの個数が減少しなくなるまで追加コスト +4 の辺を追加する．この処理によって連結なグラフが得られる．最後に，グラフ上の閉路を除去する．ここで閉路を構成する辺の除去コストは 0 以上なので，除去コスト 0 の辺を除去して閉路を解消することで回転コスト最小の全域木が得られる．

次に，アルゴリズム 2 に示されている関数 SEARCHCHANGEREGION の処理について詳述する．関数 SEARCHCHANGEREGION は与えられた線分 l の各頂点から，方向 d と直行する方向に置き換え領域に含める頂点を選択していく．そのために 8 行目では，関数 NEXTVERTICES によって，現在の置き換え領域 tmp_reg に方向 d と直行する方向で隣接している頂点の集合 $next\_vertices$ を得ている．$next\_vertices$ の中から，頂点の追加による回転コストの減少量が最大の頂点を選んで置き換え領域に含めていくことで，回転コストの減少量が最大の置き換え領域を探索している．12 行目の関数 MINADDCOST が回転コストの減少量最大の追加頂点を選択する関数である．頂点を置き換え領域に含める時の回転コストの変化は，その頂点の 8 近傍頂点の方向によって決められている．この時，置き換え領域外の水平垂直頂点は方向 d の頂点として取り扱っている．これにより本アルゴリズムは入力の格子グラフから簡略表現の特徴を利用した探索を行うことが可能となっている．また，回転コストの減少量が等しい頂点が複数存在する場合は，より線分 l に近い方の頂点を選択する．これにより必要最低限の置き換え領域でより多くの線分の置き換えを可能としている．置き換え領域探索の例を**図 7** に示す．この例は上段左端のパターンにおける頂点数 3 の垂直線分から探索を開始している．以上の処理を $next\_vertices$ が空集合になるまで繰り返す．各繰り返しにおいては，16-20 行目の処理によって回転コストの減少量が最大であり，そのコスト減少量のもとで連結なグラフの個数の減少量が最大の置き換え領域を保存している．ループ終了時の $region$，$cost$ および reg_num が，回転コストの減少量が最大の領域，回転コストの減少量および連結なグラフの個数の減少量である．

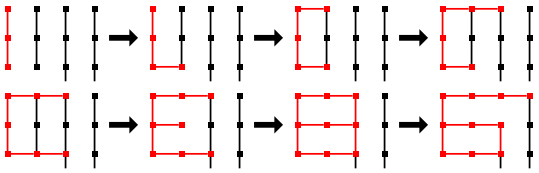

図 7 置き換え領域探索の例.
Fig. 7 Example of replacement area search.

4.3 アルゴリズムの停止性および計算量

本アルゴリズムの停止性について述べる．本アルゴリズムの全体を通して，回転コストを増加させる処理は存在しない．またアルゴリズムの処理において回転コストの変化がない区間では連結なグラフの個数は増加しない．そのため，回転コストまたは連結なグラフの個数が減少と増加を繰り返すような処理のループは発生しない．従って，本アルゴリズムは必ず停止する．

次にアルゴリズムの計算量について考察する．アルゴリズム中の各処理について，それぞれ独立に処理の繰り返し回数の最大値を考える．アルゴリズム中では，ある線分に対する置き換え領域の探索処理 (a)，集合 L に含まれる線分に対して処理 (a) を適用していく処理 (b)，領域置き換えを行う度に L を更新して処理 (b) を行う処理 (c)

という階層的な 3 つの処理および最後に実行される関数 ConnectRegs で構成されていると考えることができる．頂点数 $n*n$ の格子グラフの部分グラフ上での各処理の計算ステップ数の最大値を考える．まず処理 (a) について考える．処理 (a) については，長さ n の線分のそれぞれの頂点に対して $n-1$ 回の辺の追加を行って領域の探索を終了する場合が計算のステップ最大である．従って最大ステップ数は n^2-n である．次に処理 (b) について考える．処理 (b) は最大で集合 L の要素数と同じ回数繰り返される．頂点数 $n*n$ の格子グラフの部分グラフに対する集合 L の要素数について考える．L の要素数はグラフ上の線分の個数であり，線分の個数はそれぞれの頂点 1 つを長さ 0 の水平線分および垂直線分とみなした場合の $2n^2$ よりは必ず少なくなる．よって処理 (b) の最大計算ステップ数は $O(n^2)$ である．最後に処理 (c) について考える．処理 (c) の実行回数はアルゴリズム中で行われる領域置き換えの回数である．頂点数 n^2 のグラフ上で同時に存在し得る領域の最大数は n^2 であり，また置き換えを行った領域全体が置き換え前の状態に戻されることはないので，領域置き換えは最大でも $2n^2$ 個の各領域の置き換えの定数倍以内である．よって処理 (c) の最大ステップ数は $O(n^2)$ である．アルゴリズム全体のステップ数は処理 (a)，(b) および (c) のステップ数の乗算で求まる．ここで，各処理について独立に求めた最大値を掛けあわせると，(a)，(b) および (c) におけるステップ数は $O(n^6)$ である．関数 ConnectRegs については，頂点数 $n*n$ の格子グラフ上で最大 n^2-1 回の辺の追加および $(1/2)(n^2-n)-(n^2-1)$ 回の辺の除去が行われるため，計算のステップ数は $O(n^2)$ である．全体の計算量はこれらの和を取る．従って，本アルゴリズムは頂点数 N に対して，$O(N^3)$ で計算が終了することが保証されている．

4.4 解の最適性の証明

まず，ある簡略表現 F と最適解を与える F との差 S を考える．ここで簡略表現同士の差を，両者の間で方向が異なっている頂点の集合とする．また S の要素の中で方向が一致かつグラフ上で隣接している頂点集合を S に含まれるひとつの領域 R_S と考える．この時，以下の性質が成り立つ．

- 置き換える順番を問わず，S に含まれる領域 R_S を 1 つずつ置き換えて得られる簡略表現は，S 全体を一度に置き換えた簡略表現と等しい

ここで以下の性質が成り立つと仮定すれば，アルゴリズムで発見された置き換え領域を次々と置き換えていくことで最適解を与える簡略表現 F を得られる．

- アルゴリズムの置き換え領域探索は S に含まれる領域 R_S を全て発見する

関数 ConnectRegs は F における連結なグラフの個数が j の時に，追加コスト $+2$ および $+4$ の辺を合計 $j-1$ 個追加することで連結なグラフを構成する．その後，追加/除去コスト 0 の辺を除去することで閉路を解消して木を得る処理である．そのため，1 つの最適な簡略表現 F から得られる木の回転コストは一定であり，必ず回転コスト最小の全域木が得られる．

本アルゴリズムで得られる解が最適であることを示すために，上記の仮定を証明する．領域 R_S は同方向の線分で構成される領域である．本アルゴリズムによる置き換え領域探索は簡略表現上の線分毎に，線分に直行する方向に向けて行われるため，領域 R_S の最長の線分の長さが領域 R_S の長さと等しい場合は，この線分から領域探索を行うことでその領域 R_S 全体を含んだ範囲を探索することが可能である．この探索においては，ある線分から始まる探索範囲内で置き換え可能な線分が k 本存在する場合に，置き換える本数が 1 から k 本までのそれぞれの場合において回転数が最小となるような隣接領域との接続パターンを調べている．従って，ある線分から始まる探索範囲内で回転コストの減少量が最大の領域を発見可能である．

また，最長線分ではない線分から探索を行い，領域 R_S の部分領域が置き換え領域として発見された場合は残りの領域も置き換え領域として発見される．また同様に領域 R_S の最長線分の長さと領域の長さが異なっている場合も領域 R_S の部分集合が置き換え領域として発見され，残りの領域も置き換え領域である．従って，本アルゴリズムにおける領域探索は領域 R_S を必ず発見することができ，発見した置き換え領域を次々と置き換えていくことによって回転コスト最小の解を得ることができる．

アルゴリズム 3 回転コスト最小の全域木からの複数ロボットの動作計画の導出.

Input: 全域木 $T=(V,E_t)$，ロボット台数 N
Output: 複数ロボットの動作計画を表す全域森 P

```
1: P ⇐ {T}
2: loop
3:     (P, T_a) ⇐ SelectMaxTree(P)
4:     (T'_a, leaf) ⇐ CutLeaf(T_a)
5:     (P, T_b) ⇐ SelectOptTree(P, N, leaf)
6:     T'_b ⇐ MargeTree(T_b, leaf)
7:     P ⇐ P + {T'_a, T'_b}
8:     if LoopStopCondition then
9:         break
10:    end if
11: end loop
```

5. 複数ロボットを対象とした動作計画の決定

本章では，単体ロボットを対象として得られる全域木を利用して複数ロボットの動作計画を決定することを考える．まず，複数ロボットの動作計画の最良性について述べる．動作計画の良さを判断する基準の 1 つとして，単体ロボットの場合と同様にタスク完了までに要する時間の短さが考えられる．また複数ロボットの場合は，タスク完了ま

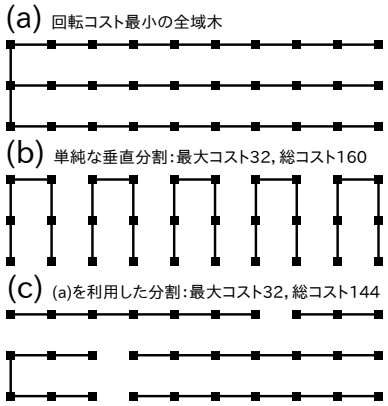

図 8 単体ロボットを対象とした最適解を利用した動作計画の例.
Fig. 8 Examples of motion plan for multi robots.

でに要する複数ロボットシステム全体での消費電力のようなコストも，動作計画の良さを判断する基準として考えられる．

次に，単体ロボットを対象として得られる全域木から複数ロボットの経路を得るためのアルゴリズム 3 の設計方針について述べる．ロボットの台数を N とする．また，複数ロボットの動作計画を N 個の木で構成される全域森 P で表現する．アルゴリズム全体の流れは，P に含まれる N 個の木に対してコストを平均化する処理である．アルゴリズム 3 の 3 行目で関数 SelectMaxTree を用いて P からコスト最大の木 T_a を取り出している．4 行目では関数 CutLeaf によって葉頂点 $leaf$ を切り出している．5 行目で $leaf$ の接続先の木 T_b を関数 SelectOptTree によって P から取り出している．この時，P の要素数が N 以下の場合は新たに頂点数 0 の木を与えている．6 行目では関数 MargeTree によって $leaf$ を T_b と接続し，7 行目で $leaf$ を取り除いた T_a および $leaf$ を接続した T_b を P に戻している．この処理を繰り返していくことで木のコストの平均化を行う．この時，関数 SelectMaxTree, CutLeaf, SelectOptTree における木および葉頂点の選択基準，ループの停止条件 *LoopStopCondition* について適切な設定を与えることでタスク完了までに要する時間が最小となる動作計画を求めるアルゴリズムが設計できると考えられる．また，このアルゴリズムは回転コストが最小の全域木から処理を開始するため，木や葉頂点の選択時に元の全域木又は簡略表現を参考にして回転コストを最小に抑えることで，**図 8** の様に複数ロボットシステム全体でのコストが少ない動作計画を得られることが期待される．図 8 では 1 回の直進移動および回転動作にかかるコストはそれぞれ 1 としている．

6. まとめ

本研究では，単体ロボットによるエリアカバレッジタスクに対して，回転動作を考慮してカバレッジ時間が最小となるような経路を与える全域木の探索アルゴリズムを提案した．また，単体ロボットを対象として提案したアルゴリズムが頂点数 N に対して 3 次の多項式時間以内で探索を終了し，最適解を求められることを示した．これは，指数時間の計算が必要となる単純な全探索の手法と比較して高速である．さらに，単体ロボットを対象とした最適解を利用する複数ロボットの動作計画アルゴリズムについての検討を行った．単体ロボットを対象とした回転数最小の木を利用することで，ロボットシステム全体でのコストが最小化されることが期待される．

今後の課題としては，複数台ロボットによるエリアカバレッジタスクのための動作計画アルゴリズムの設計やその評価が必要であると考えられる．

謝辞 本研究の一部は JSPS 科研費 18K18024 の助成による．

参考文献

[1] Gabriely, Y. and Rimon, E.: Spanning-tree based coverage of continuous areas by a mobile robot, *Annals of Mathematics and Artificial Intelligence*, Vol. 31, No. 1, pp. 77–98 (2001).
[2] Bochkarev, S. and Smith, S. L.: On minimizing turns in robot coverage path planning, *Proc. of Int'l Conf. on Automation Science and Engineering (CASE)*, pp. 1237–1242 (2016).
[3] Schfle, T. R., et al.: Coverage path planning for mobile robots using genetic algorithm with energy optimization, *2016 International Electronics Symposium (IES)*, pp. 99–104 (2016).
[4] Jimenez, P. A., et al.: Optimal area covering using genetic algorithms, *Proc of Int'l Conf. on advanced intelligent mechatronics*, pp. 1–5 (2007).
[5] Yakoubi, M. A. and Laskri, M. T.: The path planning of cleaner robot for coverage region using Genetic Algorithms, *Journal of Innovation in Digital Ecosystems*, Vol. 3, No. 1, pp. 37 – 43 (2016).
[6] Kapanoglu, M., et al.: A pattern-based genetic algorithm for multi-robot coverage path planning minimizing completion time, *Journal of Intelligent Manufacturing*, Vol. 23, No. 4, pp. 1035–1045 (2012).
[7] Màntaras, L., et al.: Generation of Unknown Environment Maps by Cooperative Low-cost Robots, *Proc. of the 1st Int'l Conf. on Autonomous Agents*, pp. 164–169 (1997).
[8] Ryu, S.-W., et al.: A search and coverage algorithm for mobile robot, *Proc. of 8th Int'l Conf. on Ubiquitous Robots and Ambient Intelligence (URAI)*, pp. 815–821 (2011).
[9] Hazon, N. and Kaminka, G. A.: Redundancy, Efficiency and Robustness in Multi-Robot Coverage, *Proc. of Int'l Conf. on Robotics and Automation*, pp. 735–741 (2005).
[10] Zheng, X., et al.: Multi-robot forest coverage, *2005 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 3852–3857 (2005).
[11] Shrock, R. and Wu, F. Y.: Spanning trees on graphs and lattices in d dimensions, *Journal of Physics A: Mathematical and General*, Vol. 33, No. 21, p. 3881 (2000).