車両型ロボットの移動に関する環境に適応した動作制御手法の研究

岩本, 憲泰

https://doi.org/10.15017/1654871

出版情報：九州大学, 2015, 博士（工学）, 課程博士
バージョン：
権利関係：全文ファイル公表済

車両型ロボットの移動に関する
環境に適応した動作制御手法の研究

工学府
機械工学専攻
岩本　憲泰

目 次

第1章　緒言

1.1　移動ロボットと未知環境での移動

我々人間は，目標位置が与えられた場合，さまざまなことを考えながら移動を行う．何も障害物が存在しない平面上であれば，非常に簡単であるが，実際には多くの障害物が存在するため，障害物を認識した上で，姿勢を考慮した軌道を計画しなければならない．また，目標位置までの経路が未知であるならば，地図を検索しながら，もしくは頭の中で地図を構成しながら移動を行う．経路の途中に水たまりがあった場合には，飛び越えるか迂回するか，そのまま水たまりの中に入ってしまうか等の移動方法を考える．つまり，移動とは複数の技術の組み合せで成り立っている．

ロボットが移動を行う場合にも図 1.1 に示すような同様の技術（環境の認識，環境地図の生成，自己位置の推定，軌道のプランニング，軌道の追従）が必要になってくる．移動ロボットには脚型，車両型，クローラー型など様々な形態が存在するが，基本的に必要な技術は同じであり，重心を移す方法が異なる．本論文では，機構と移動のための制御が簡便で，最も基礎となる移動ロボットと考えられる車両型ロボットに関して焦点を当てて議論を行う．

1.1.1　環境に適応した動作制御

先に述べた通り，ロボットの移動には図 1.1 に挙げたような基本技術が必要となる．さらに，ロボットがある動作を行う場合には，ロボットを制御する手法である動作制御手法が必要となる．未知の環境では，様々な状況が想定され，既知環境に比べて，ロボットに要求される動作制御手法のレベルは高くなる．

未知の環境では，ロボットが動作することで徐々に環境の情報が明らかになって

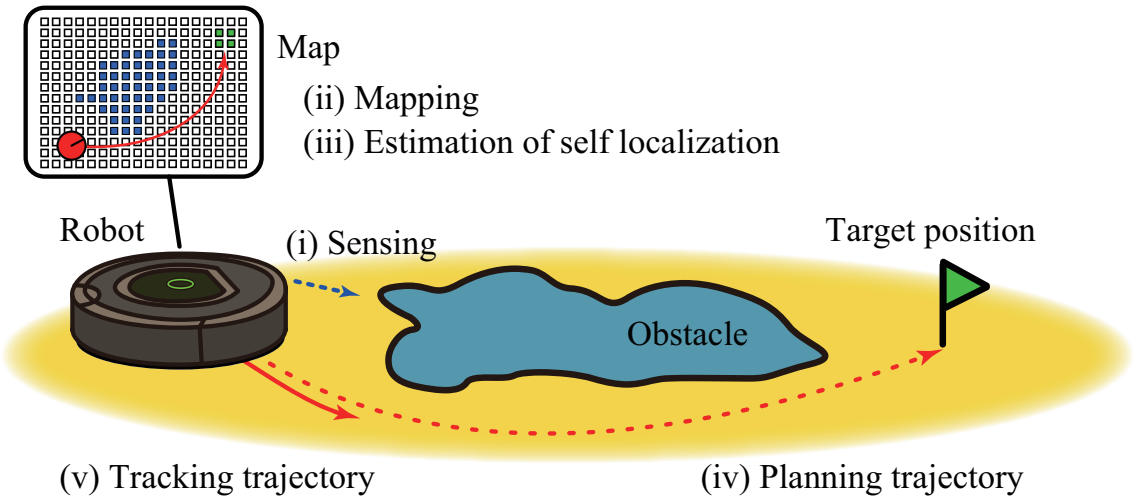


Fig. 1.1 A mobile robot can moves by using multiple techniques as shown in this figure (i)-(v).

くる．ロボットが地図生成と自己位置推定を同時に行っているとすると，地図が大きく修正されることも珍しくなく，ロボットがおかれる状況は時々刻々と変化する．ロボットがあるタスクを実行しているとすると，変化する状況に合わせて動作制御手法を切り替える手法が必要となってくる．また，未知環境や不整地環境での移動中には，ロボットの移動を妨げるような事態に，ロボットが遭遇することも考えられる．例えば，ロボットの転倒，移動経路の断裂，浸水などが挙げられるが，ロボットはこれらの事態に対処するため，従来よりも高度な動作制御手法が必要となってくる．本稿では，上記のような 2 つの環境に適応した動作制御手法を提案する．本稿で述べられている結果は，すでに文献 [1]-[3] に掲載されている．

1 つ目には，現在最も普及しているロボットである掃除ロボットに関する動作制御手法である．掃除とは，閉じた 2 次元領域を覆い尽くす作業と一般化でき，これらの作業は掃引と呼ばれる．未知環境の掃引は，オンラインでの地図生成と同時に行われ，地図が更新されるごとに既掃引領域（既に掃引した領域）か未掃引領域（未だ掃引していない領域）かの正しい判断を行うことができる．掃引初期の地図は信頼性が低く，頻繁に大きく修正を繰り返すため，掃引経路を頻繁に再計画する必要がある．一方，掃引終盤では地図は信頼性が高く，全体の地図情報を用いた経路計

画が有効になってくる．また，ロボットは移動することで自己位置の確からしさが低下する．自己位置推定の結果と実際の位置が大きく違うと，掃引を行っても意味がない．このように，そのときの状況に応じて，ロボットは動作制御の切り替えを行う必要がある．しかし，適切な動作制御の切り替えを行わなければ，環境によっては掃引効率の低下を招く恐れがある．動作環境の大きさや形状に関係なく，掃引が進むにつれ，選択される動作制御の割合が同じ傾向となるアルゴリズムが望まれる．このような掃引動作制御手法を 2 章で述べる．

2 つ目には，車両ロボットのための不整地で移動能力を向上させる研究について述べる．車両ロボットは，平地での移動効率が高いが，不整地環境では数多くの移動を妨げる事象が発生する．特に，不整地の斜面や路面の凹凸は，復帰不可能な転倒状態に陥る可能性が高い．また，計画した経路が実際に移動可能とは限らず，予期しない路面の断裂や浸水により，計画した経路を変更せざるを得ない状況も考えられる．動物界では，尻尾は様々な場面で用いられており，ロボットの分野でも様々な事態に対処するための機構として，利用可能であると考えられる．3 章では，尻尾という機構を車両ロボットに搭載し，路面の凹凸による移動不能状態を，未然に回避するための跳躍動作を提案する．また，跳躍後の着地点位置や姿勢は，その後の転倒につながることから重要となる．そこで，跳躍することで目標位置姿勢に到達するための動作制御手法について提案する．

本章では，まず従来までの移動ロボットに関する技術を次節でまとめ，次に掃引ロボットに関する研究，尻尾に関する記述やロボットへの応用研究についてまとめる．

1.2　移動ロボットの移動に関する技術

移動ロボットの図 1.1（i）から（v）のような基礎的な技術に関する研究は数多く行われている．本稿において利用する技術や関連した技術について紹介する．

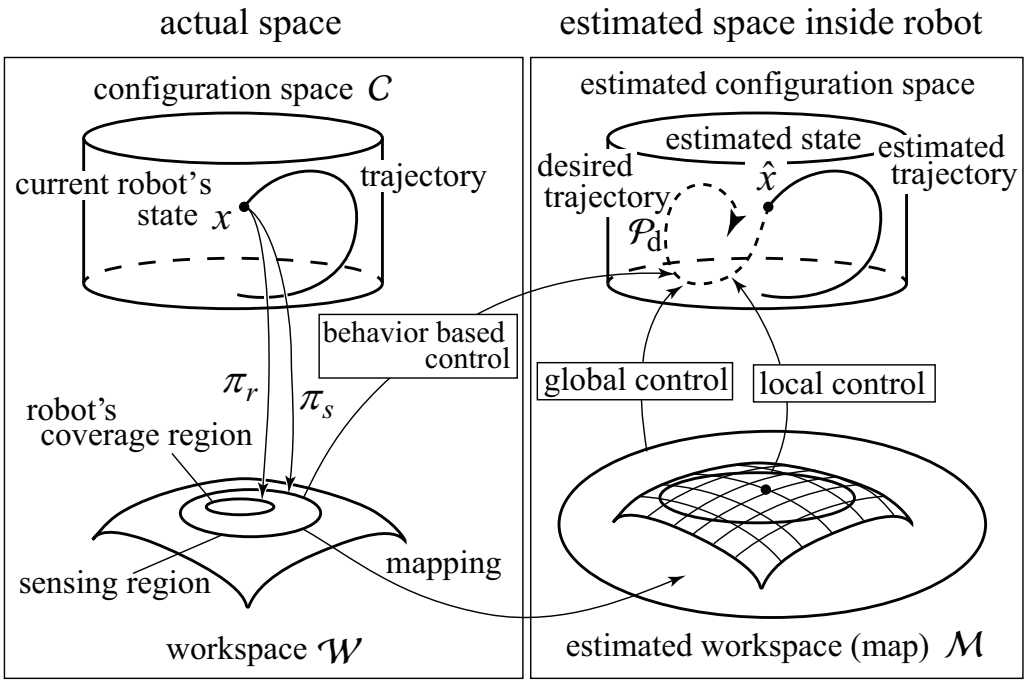


Fig. 1.2 The schematic of motion control and motion planning of mobile robots

1.2.1　動作制御と軌道計画

軌道計画の概念図を図 1.2 に示している．以下の記述は，文献 [4] や [5] に詳しく記載されている．ロボットの位置姿勢空間は C-Spase と呼ばれ，ある位置姿勢は C-Space 上の一点で表現される．位置と姿勢空間であるため，C-Space はベクトル空間と考えることができるが，一方で基準点からの同次変換行列空間とみることもでき，これは Lie 群にあたる．つまり，平面上を移動する一つの剛体ロボットを考えたとき（環境 $\mathcal{W}$ が $\mathcal{W} \in \mathbb{R}^2$ のとき），C-Space は $\mathcal{C} = SO(2) \times \mathbb{R}^2 = SE(2) \in \mathbb{R}^3$ と表せる．ちなみに，ロボットの速度角速度は，リー環 $\mathfrak{se}(2) \in \mathbb{R}^3$ で表される．この C-Space のある状態 x から，$\mathcal{W}$ 上にロボットが占有する領域と，ロボットに搭載されているセンサが環境を認識している領域上に移す写像 π_r, π_s が存在する．$\mathcal{W}$ には，障害物領域 $\mathcal{O}$ が含まれるており，同じ位置であっても姿勢によっては，障害物によりロボットは存在することができない．ロボットは，C-Space から障害物によ

りロボットが存在できない状態を引いた空間$\mathcal{C}_{free}$のみで存在できる．障害物領域を差し引いた空間はC_{free}と表記され，

$$\mathcal{C}_{free} = \{x \in \mathcal{C} | \pi_r(x) \cap \mathcal{O} = \Phi\} \tag{1.1}$$

と表される．ここでΦは空集合を意味している．ロボットの軌道計画はこのC_{free}空間内で行わなければならない．ロボットがC_{free}空間で軌道計画を行うためには，環境$\mathcal{W}$が既知という前提が必要であり，環境$\mathcal{W}$が既知でない場合には，$\mathcal{W}$を推定する必要がある．過去の自らの状態とセンサ情報から得られた情報$\pi_s(x)$から現在の自らの状態$\hat{x}$を推定するのが，自己位置推定手法であるのに対し，センサ情報から得られた情報$\pi_s(x)$と$\hat{x}$を用いて，環境全体を推測する手法が地図生成手法である．この推定された環境を地図$\mathcal{M}$と呼ぶ．$\mathcal{W}$が既知でない場合，ロボットは地図$\mathcal{M}$を用いて軌道計画を行うので，$\mathcal{M}$で障害物と考えられる領域$\hat{\mathcal{O}}$とロボットが重複する状態を除いた空間

$$\hat{\mathcal{C}}_{free} = \{x \in \mathcal{C} | \pi_r(\hat{x}) \cap \hat{\mathcal{O}} = \Phi\} \tag{1.2}$$

の中で軌道$\mathcal{P}_d$の計画を行う．

未知環境内でロボットが現在位置から目標位置に移動することを考えたとき，軌道計画を行うには複数の手法が存在する．一つ目は，センサ情報から得られた情報$\pi_s(x)$から目標軌道$\mathcal{P}_d$を求める方法である．これは，Behavior based 手法であり，センサの検出範囲までの軌道計画しか行うことができない．一般的には軌道計画というよりも，センサからの情報を基にロボットをどのように制御するかという意味で動作制御と呼ばれる．二つ目は，ロボットの周辺の地図情報を用いて目標軌道$\mathcal{P}_d$を求める手法で，これを本稿では Local Reference based 手法と呼ぶ．周辺の地図情報と目標値までの方向を用いた手法がよく用いられる．計算量は少ないものの，周辺の情報だけであるため，簡単なアルゴリズムでは最適な経路でない場合や目標値に到達できない場合も存在する．三つ目は，地図全体の情報を用いて目標軌道$\mathcal{P}_d$を求める手法（Global Reference based 手法）で，現在の位置姿勢から目標値までの経路計画を行う．最適な経路を求めることが可能であるが，計算コストが非常に大きいため，準最適な経路を求める手法がよく用いられる．

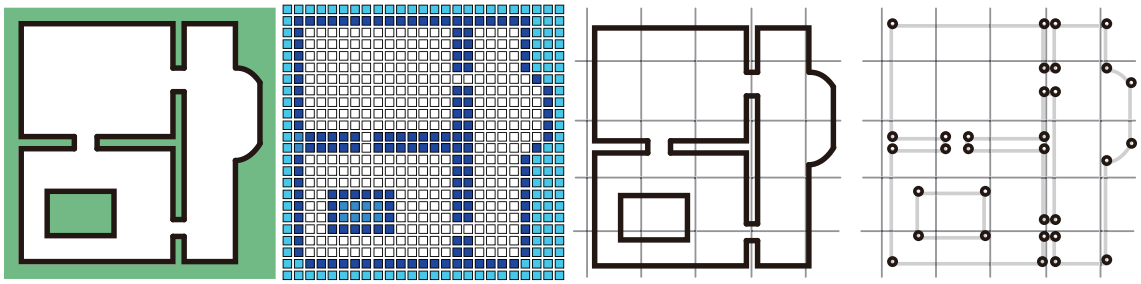


Fig. 1.3 Coordinate-Based Representation Map

1.2.2　環境地図表現手法

地図はロボットの移動や軌道の計画に重要な要素の一つである．地図は，環境の情報を間引いてロボット内に格納するというとらえ方ができ，地図の分解能が低いほど計算量・情報量が小さいため，計算コスト・格納コストが低いが，移動や軌道計画には不利になる．Wallgrun[6] の地図の分類の仕方を参考にすると，座標系内で定義される地図（Coordinate-based representation）とつながりだけで定義される地図（Relational representation）があり，両方を組み合わせて用いられることが多い．Coordinate-based repr. のみで経路計画を行うと計算量が多い．そこで，ロボットが通過できる範囲を認識するために，Coordinate-based repr. と Relational repr. とを同時に作成する方法，もしくは，Coordinate-based repr. 内の障害物と推定される領域を除いて分割した領域群から Relational repr. を生成する方法が用いられる．

図 1.3（a）の環境を表現した，3 種類の座標系内で定義される地図を図 1.3（b）～（d）に示している．ある絶対座標系の中で，構成要素にある座標を位置付けるもので，図 1.3（b）～（d）の順に，格子上にその位置が障害物である確率を表す手法，点や線などにより障害物を表す手法，ランドマークの位置情報のみを持つ手法である．座標系を持たず，構成要素間のつながりだけで定義される地図を図 1.4（b）～（c）に示している．丸はノードを意味し，ノード間を結ぶ線がエッジを意味している．図 1.4（b）は View Graph[7][8] と呼ばれ，ノードは特色あるセンサ情報により生成される．一方，図 1.4（c）は Route Graph[9]-[11] と呼ばれ，ドアや廊下など環

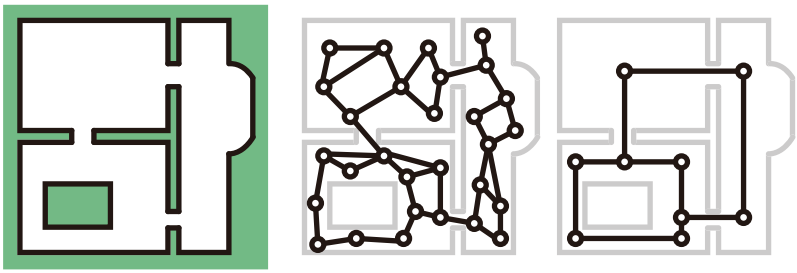


Fig. 1.4 Relational Representation Map

境の特色ある位置からノードを構成する手法を表している．

この中でも，Elfes[12][13] と Morvac[14] により提案された Occupancy Grid Map（図 1.3（b））について触れておく．Occupancy Grid Map は，占有格子地図とも呼ばれ，座標系内の格子状のセルは占有領域，自由領域，未知領域の 3 つの状態を表現することができる．特に，確率的な表現が可能であり，後述する Kalman Filter との相性が非常に良い．Thrun による文献 [15] や [16] に詳しくまとめられている．

環境の地図 $\boldsymbol{m}$ はそれぞれのセル $m_i \in \boldsymbol{m}$ に分割され，各セルがフリー領域，障害物領域，未知領域であるかを占有確率と呼ぶ数値で表現する．セル m_i は，他のセルとの独立であると仮定されており，$\boldsymbol{m}$ となる確率 $p(\boldsymbol{m})$ は

$$p(\boldsymbol{m}) = \prod_{m_i \in \boldsymbol{m}} p(m_i) \tag{1.3}$$

で表される．時刻 0 から t のロボットの姿勢 $\boldsymbol{x}_{0:t}$ とそのセンサ観測 $\boldsymbol{z}_{0:t}$ をするとき，時刻 t でのセル m_i の占有確率を $p(m_i|\boldsymbol{z}_{0:t}, \boldsymbol{x}_{0:t})$ と表す．占有確率を $p(m_i|\boldsymbol{z}_{0:t}, \boldsymbol{x}_{0:t})$ はベイズの定理を用いると

$$p(m_i|\boldsymbol{z}_{0:t}, \boldsymbol{x}_{0:t}) = \frac{p(\boldsymbol{z}_t|m_i, \boldsymbol{z}_{0:t-1}, \boldsymbol{x}_{0:t})p(m_i|\boldsymbol{z}_{0:t-1}, \boldsymbol{x}_{0:t})}{p(\boldsymbol{z}_t|\boldsymbol{z}_{0:t-1}, \boldsymbol{x}_{0:t})} \tag{1.4}$$

とできる．上式に確率 $p(x)$ のオッズ

$$\mathrm{Odds}(x) = \frac{p(x)}{1 - p(x)} \tag{1.5}$$

を導入し，確率が 0，1 付近の場合に確率計算の不安定さを回避するため対数オッズ $\mathrm{logOdds}(x)$ を用いて，式を変形すると

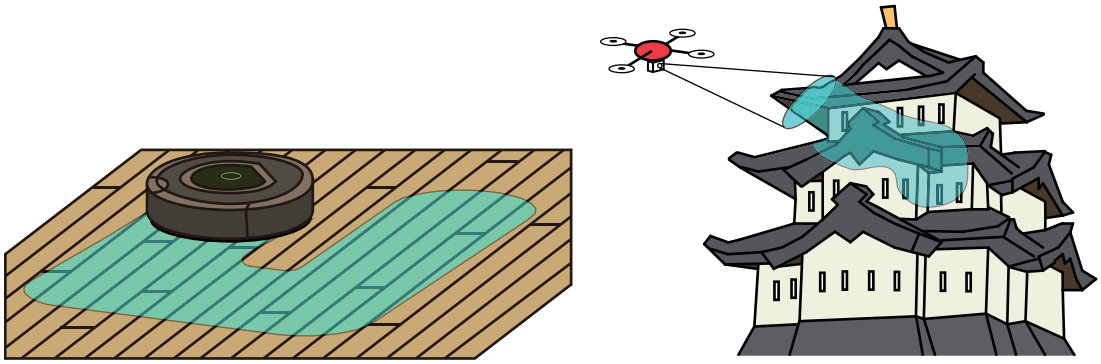
$$\begin{aligned}\mathrm{logOdds}(m_i|\boldsymbol{x}_{0:t}, \boldsymbol{z}_{0:t}) = \mathrm{logOdds}(m_i|\boldsymbol{z}_t, \boldsymbol{x}_t) \\ - \mathrm{logOdds}(m_i) + \mathrm{logOdds}(m_i|\boldsymbol{x}_{0:t-1}, \boldsymbol{z}_{0:t-1})\end{aligned} \tag{1.6}$$

が得られる．ここで，各セルの初期占有確率は 0.5（未知）と設定すると，$\mathrm{logOdds}(m_i)$ は 0 となる．そのため，地図の値を対数オッズで表現することで，現在の値（左辺）は，一つ前の値（右辺 3 項目）に，現在のセンサ情報から求められる値（右辺 1 項目）を足すのみで得ることが可能となる．$\mathrm{logOdds}(m_i|\boldsymbol{z}_t, \boldsymbol{x}_t)$ には，時刻 t での観測と姿勢に基づく占有確率 $p(m_i|\boldsymbol{z}_t, \boldsymbol{x}_t)$ を定義する必要がある．一般的にセンサ観測の不確かさがその値を計算するために考慮される．

1.2.3　自己位置推定手法

ロボットの自己位置を推定するためには，ロボットのモデル（状態方程式）から状態の時間変化を算出し，次の状態を推測するというのが最も簡単な方法である．しかし，状態方程式には誤差が含まれるため，推定された状態と現在の実際の状態との誤差は，時間が経つにつれ大きくなっていく．そこで，ロボットのセンサから得られた観測情報と推定値を組み合わせることで，良い推定値を得られるのではないかと考える．

これが，有名な Kalman Filter[17][18] と Particle Filter[19][20] である．この 2 つのフィルタは，これまでの実績から信頼度が高いため，よく用いられる．どちらもセンサ情報や制御における誤差をモデル化し，自己位置に尤度を適応している点で，確率的な手法と言える．文献 [16]，[21]，[22] は非常に詳しく，これらについてまとめてある．正確に述べると，Kalman Filter は線形な状態方程式の場合のみを言い，非線形な状態方程式の場合には，テイラー展開を用いて線形化した Extended Kalman Filter（EKF）[18] と言う．この Kalman Filter と Particle Filter の大きな違いは，Kalman Filter は平均と共分散行列というパラメータを更新していくのに対して，Particle Filter は全てのパーティクルの状態を更新していく点である．Kalman Filter はパラ



(a)　(b)

Fig. 1.5 Sweeping

メータだけを逐次的な計算で引き継ぐため，ロボットの状態のみを推定する場合に限ると，計算量としては Particle Filter の方が圧倒的に大きいことを意味する．このようにパラメータを更新していく Kalman Filter のようなフィルタは Parametric Filters，パラメータを持たない Particle Filter のようなフィルタは Nonparametric Filters と呼ばれる [16].

地図の生成は，センサ情報に加え，自己位置の推定値も用いられる．そのため，自己位置推定問題と地図生成問題は切り離すことのできない．これが，Simultaneous Localization and Mapping (SLAM)[21][22] と称される問題である．特に，自己位置推定の確率的手法と地図生成の Occupancy Grid Map は非常に相性がよい．

1.3　掃引作業と探索

ロボットが閉じた 2 次元の領域を覆い尽くす作業は掃引作業（sweeping）と呼ばれる（図 1.5（a）, (b)）．応用範囲としては，掃除ロボット，塗装，草刈り，地雷探索，建物の検査と幅広い．切削のためのツールパスや 3D プリンタのヘッドのパスを生成する手法もこれにあたる．特に環境の地図を生成することや，環境を把握することを目的とした作業は探索（exploration）と呼ばれる．これらの領域を覆い尽くす問題を総称して Complete Coverage Problem （CCP）と呼ぶ.

近年では，家庭用掃除ロボットの掃引動作の効率性が注目されている．2002 年に iRobot 社が Roomba[23] を発売し，これまでに多くの掃引ロボット [24]-[29] が各社から発売されている．カメラ，レーザーレンジファインダやランドマークを使用し，自己位置推定や地図生成を行う掃引ロボットは存在するものの，アルゴリズムに関しては完全には公開されていない．明確ではないが，上記のロボットが効率的な掃引を行うには，環境の限定や人による暗黙の介入が必要であるように見受けられる．掃引に関する問題は，ロボットの動作に関する問題と，通過時にどの程度ゴミを吸い取れるか等の機構の能力に関する問題とに分けることができる．本稿では，動作に関する問題にのみ着目する．

図 1.2 を用いると，ロボット本体や手先等の部位によって覆われた領域 $\pi_r(x(t))$ の和集合が作業空間 $\mathcal{W}$ と等しいとき，掃引が終了したと言えるため

$$\mathcal{W} = \bigcup_{t=0}^{T} \pi_r(x(t)) \tag{1.7}$$

となる有限時間 T が掃引終了時刻となる．掃引に関する研究では，いかに T を小さくするかを目指しており，一般的に掃引効率が良い動作制御アルゴリズムとは，この T が小さいアルゴリズムを言う．一方，図 1.2 でロボットが環境を覆う領域 $\pi_r(x(t))$ を集めるのではなく，センサが観測した領域 $\pi_s(x(t))$ を集める作業の場合は，有限時間 T で

$$\mathcal{W} = \bigcup_{t=0}^{T} \pi_s(x(t)) \tag{1.8}$$

となるとき，作業が完了したことを意味する．

1.3.1　過去の掃引に関連する研究

掃引作業の計画手法，動作制御手法に関する研究は，地図を利用しない手法，与えられた地図を用いて計画する手法，地図を生成しながら動作する手法に分かれる．

環境の地図を利用しない手法は，掃引作業において掃引の完了を判定できず，アルゴリズムは掃引完了を保証しない．しかし，動作制御に関する計算量は非常に少なく，経験的にある程度の掃引効率となることは保証されている．地図を生成せず

に動作する市販の掃除ロボットは，反射動作や壁追従動作，螺旋動作を基本として動作する．Palacin ら [30] は掃除ロボットの動作解析を行うカメラシステムを示し，Palleja ら [31] はこのシステムを用いることで，市販の掃除ロボットの動作を解析し，掃引率のモデル化を行った．

カオスとは初期値鋭敏性と位相推移性を同時に満たす系のことを言い，位相的推移性を利用することで掃引作業につなげる研究もある．中村ら [32] はロボットの運動に Arnold 方程式を取り入れることで，Fallahi ら [33] は複数のロボット間で協調した掃引動作を行うための手法を提案した．荒木ら [34][35] は，矩形領域内では確率的な反射動作を行わずとも，あらかじめ決められた角度を用いた壁反射動作だけで，領域内の掃引が可能であることを示した．矩形領域内のある断面（水平線）を取り出し，ロボットが通過する断面上の位置を数列として表現し，この数列の（リアプノフ指数を用いた）位相的推移性を掃引の完全性として評価した．

ロボットに環境の地図を与え，与えられた地図を基に掃引経路を計画する問題に対し，地図を分割し，分割された領域を基に掃引経路の計画を行う手法が多く提案されている．以下では，分割された領域をセルと呼ぶ．セルがロボットの大きさよりも小さな基本図形であるとき，ロボットはすべてのセルを巡る問題となる．この問題に関して，Zelinsky らは，四角のセルに対して Distance Transform 法を使った掃引アルゴリズムを提案している [36]．Oh らは，四角形のセルでなく三角形のセルを用いることで，距離の短い経路を生成することを提案した [37]．Gabriely らは，ロボットのサイズを D としたときに，$2D$ サイズのセルで掃引環境を分割し，セルの中心を結ぶ仮想的な壁を設定する Spanning Tree Covering (STC) アルゴリズム [38] を提案している．ロボットはこの仮想的な壁に沿って移動することで，セルを四分割した領域全てを通過することができる．

次にセルがロボットよりも大きく，環境の形状に合わせて分割される場合を考える．セルが凸領域であるとき，ロボットはセル内を一筆書きで掃引することができる．すると，セル内の掃引経路計画とセル間の移動経路計画に分けることができる．Choset は，ジグザグ経路で掃引が可能なセルに分割する，Boustrophedon Celleular Decomposition を提案した [88]．

地図とは環境の情報を間引いて表現されたものであるため，環境と地図の間には誤差が生じてしまう．ロボットが完全な環境情報を有した状態で計画を行えることはなく，環境と地図との誤差を検知した際には，計画した軌道から変更を強いられる．この問題に対して，倉林は地図誤差に対して反射的な動作で対応する方法を提案している [39]．深澤らは，既知の地図に対してオフラインで経路を計画し，未知障害物に遭遇した場合にオンラインで再計画する手法を提案した [40]．この手法では，オフライン時に，センサの観測可能な領域で環境を覆い尽くすよう，観測点を配置し，これらに対して巡回セールスマン問題を適用することで経路を得る．経路移動中に未知障害物に遭遇した場合には，障害物内に位置していた観測点を削除し，巡回セールスマン問題を解きなおすことで，経路の修正を行う．

地図をロボットに与えるというのは，事前の環境の測定を要し，非常にコストがかかる．また，人が生活する環境にて掃引を行う場合，イスや机などの固定されていない家具や人等の移動障害物が存在するため，与えられた地図と環境の構造は大きく異なるかもしれない．そのため，ロボットが地図を生成しながら掃引を行う状況を考えた研究というのも行われてきた．ジグザグ動作の掃引と，センサーベースによるトポロジカルマップの生成を同時に行う手法を，Acar ら [41] や Wong ら [42] が提案している．2 つの違いは，トポロジカルマップのランドマークの違いにあり，Acar らの手法では，Morse 関数の臨界点（critical points）を，Wong らの手法では環境のコーナーをランドマークとして定義している．Gabriely らは，文献 [38] にてスパンニングツリーをオンラインで生成する手法も提案している．Luo らは，ニューラルネットワークを用いて障害物を避けながら未掃引領域へ向かう経路生成手法を提案した [43]．

探索に関する最も有名な手法は Yamauchi[44] が提案した Frontier-based 手法である．この手法は，地図上で既知な領域と未知な領域の境界を Frontier と呼び，ロボットから最も近い Frontier を目指して移動を行う手法である．他にも，未知領域の大きさを情報量（エントロピー）として定義し，エントロピーの大きな方向を目指し移動する手法も挙げられる．Wattanavekin ら [45] は，深澤らが提案した手法 [40] を未知領域の探索に適用している．Stachniss[46] は，探索において評価関数を用いて，ど

の点に向かって移動するかを決定している．探索の戦略として，ロボットから最も近い点に向かう手法（CL），最もエントロピーが多く得られる点に向かう手法（IG），IG に加えオーバーラップまで考慮した手法（IG_WIN），CL と IG を組み合わせた手法（IG_CL）の 4 つの評価関数を挙げている．括弧内部の表記は，文献に記載してある呼称である．IG_CL に用いられた評価関数は，IG と CL の評価関数を重みづけしたもので，戦略を切り替えているわけではない．これらの手法は，地図全体を考慮した手法であるため，Global Reference based 手法に分類される．

1.3.2　掃引作業の問題点

上記の Complete Coverage Problem に関する研究は，ロボットの自己位置推定が完全であること，ロボットの制御量に対して移動の誤差が存在しないことが前提である．しかし，センサー誤差のない状況は存在せず，安価なセンサシステムでは移動により自己位置推定誤差が大きくなる．また，タイヤのすべり，風や波などの外力により，ロボットの意図した移動ができない状況も考えられる．上記の研究は，これらに対処することはできない．

この問題に対して，上田ら [47] は，位置推定とそれに基づいた動作制御手法を提案した．これは，Behavior based 手法と Local Reference based 手法を組み合わせたものである．上田らは Behavior based 手法を 2 つの目的で用いている．一つ目は地図誤差や位置誤差に対応するための反射的動作．二つ目は自己位置修正のためのランドマークを検出を目的とした壁追従動作である．しかし，反射的な動作は，地図の情報量がほとんどない場合の Local Reference based 手法に等しい．さらに，掃引が進むにつれ，環境の大域構造が複雑な場合には Local Reference based 手法では効率が悪い．

また，Stachniss らは探索中にロボットの位置に関する不確かさが大きな場合を考慮したアルゴリズムの提案を行っている [48]．ロボットは自己位置の不確かさを監視し，探索を継続するか，不確かさの改善を行うかを選択する．Stachniss らの手法では，ロボットは軌跡を故意に閉じる（一度通過した点を再度通過する）ことで，軌跡を修正し，不確かさの改善につなげている．効率が低下するため，既に通過した

点を通ることは好ましくないが，ロボットの位置誤差が無視できない場合には，このような動作を取り入れると，正確な探索を行えることを示している．

本稿では，ロボットのセンサ誤差や移動誤差が無視できない場合での掃引を想定したアルゴリズム動作制御手法を提案する．

1.3.3　提案する掃引動作制御手法

ロボットのセンサ誤差や移動誤差が無視できない場合に，オンラインで生成される地図は，掃引初期から掃引終盤にかけて，徐々に信頼性を高めながら完成する．そのため，掃引終盤で効率が良い地図全体を利用した動作計画は，掃引初期には無駄になることが多い．また，センサ誤差や移動誤差が無視できない状況では，ロボットの自己位置推定誤差は移動に合わせて大きくなる．自己位置推定誤差が大きい場合には，計画した経路を追従することは困難であり，経路を計画することすら無意味であるため，自己位置の確からしさを高める動作を組み合わせることが必要となる．

そこで，掃引時間が経つにつれ，動作制御が選択される割合が徐々に変化する掃引アルゴリズムを提案する．本アルゴリズムには 3 つの動作制御と 2 つの条件分岐で構成されている．3 つの動作制御は，Local Reference based 手法，Global Reference based 手法と自己位置の確からしさを高める手法．2 つの条件分岐は,「ロボット周辺の未掃引領域の量」,「ロボットの自己位置の確からしさ」に関するもので，2 つの条件分岐に従って，各動作制御を選択する．2 つの条件分岐には，掃引の終了時刻や部屋環境の大きさに関する情報が入っていないにも関わらず，掃引開始から掃引終了にかけて徐々に Local Reference based 手法から Global Reference based 手法へと選択する割合が切り替わる点が，本提案アルゴリズムと従来の手法とで大きく異なる．地図の完成度が高まるにつれ，ロボット周辺に未掃引領域が存在しない確率が大きくなるため，環境の大きさに依存することなく，掃引初期には Local Reference based 手法が主に，掃引終盤では Global Reference based 手法が主に選択されると考えられる．さらに，自己位置の確からしさは時間に依存しないとすると，自己位置の確からしさを高める手法は一定の割合で選択されると考えられる．

また，ロボットのセンサ誤差や移動誤差が無視できない状況では，計画した経路

を正確に追従することができない場面や，計画した経路自体が適切でない場面が存在する．これらの場面により掃引の後半では，未掃引領域は複数の未掃引領域の塊として存在することとなる．この問題は，従来の掃引に関する研究ではなされてこなかった．本稿では，この問題に関する議論を行い，最適な経路計画手法よりも計算量の少なく，掃引効率に影響しない準最適な経路計画手法を提案する．

本稿の掃引動作制御手法に関する結果は，既に文献 [1] と文献 [2] に掲載されている．

1.4　多目的に利用される尻尾

生物学上の尻尾とは,「動物の後端に位置する付属物」[49] で，尻尾の構造（筋肉の太さや尾椎の数等）は，ネコやイヌを見てわかる通り，様々である．動物は，各々が必要な場面で尻尾を用いており，尻尾の構造は少なからず用途に関係している．いかに尻尾が様々な場面で利用されているかを示すため，本節で尻尾に関する記述や研究について述べる．

1.4.1　動物界での尻尾

尻尾はバランスをとる役割としてよく知られている [49]．その中でも，チーターは，高速走行時のバランス取りや，急激な方向転換時の浮遊した体の姿勢修正に利用されていると考えられている．ヤモリやトカゲは空中で（落下時や跳躍後に）姿勢を修正するために尻尾を用いている．

尻尾を手の代わりとして利用する動物として，クモザル，キノボリヤマアラシが挙げられ，尻尾の先端付近を木の枝に巻き付けることで，体の落下防止に用いている．同様の利用法として，タツノオトシゴは海藻に尻尾を巻き付け，海流にさらわれることを防止する．タツノオトシゴの尻尾の構造は，複数のパーツで四角柱を構成している [50]．各パーツはばね要素で接続されており，外部から圧縮方向に力が加わると変形する．海藻のような固定物に巻き付く際には，尻尾はそれに倣うように変形し，大きな巻き付け力を発揮する．Poter らは，これを模倣した機構を開発した [50][51]．

カンガルーの尻尾の根元は，他の動物に比べ非常に太く，筋肉の発達がみられる．そのため，利用する場面が非常に多い．例えば，直立する場合，ゆっくりと移動する場合など，それぞれの状況において尻尾の利用が異なる．直立する場合には尻尾を自重の支えとして用いており，これはマングースなどの他の動物にも見られる．ゆっくりと移動する場合には，尻尾を 5 本目の脚として利用することが知られており，O'Connor らはこれをまとめている [52].

移動手段が尻尾である生物として，蛇やウナギ，ドジョウなどが挙げられる．さらに，水中に生息する哺乳類，魚類は尾ひれにより，遊泳の推力を得ている．尾ひれの形状には複数の種類が存在し，形状と遊泳能力には相関があることが知られている．海イグアナは，尻尾の形状が陸上生物のそれと全く同じでありながら，水中を泳ぐために利用している．

これまでは，現存する生物の尻尾に関する記述を挙げてきた．絶滅した恐竜にまでさかのぼると，尻尾の骨形状に，現存する生物にはない特徴を持つものも存在し，恐竜学の研究者達は，化石から彼らがどのように尻尾を利用していたかを長年議論を続けてきた．非常に長い首と尻尾で有名なのは，竜脚類に属するもので，ブラキオサウルス等がよく知られており，長い首と尻尾をシーソーのようにバランスをとるのに利用されていたと考えられている [53]．これは，四足で歩行する恐竜だけでなく，ティラノサウルスのような二足で歩行する恐竜にも言える．過去の恐竜の資料にあるティラノサウルスの図では，カンガルーのような姿勢で描かれているが，現在の資料では頭部と尻尾をつり合わせるような姿勢で描かれている [53]．文献 [53] には，ティラノサウルスの前脚が異常に小さいのは，巨大な頭部と尻尾をつり合わせなければならなかったために違いないと記載されている．

一部の恐竜の骨に見られる特徴の一つとして，骨化腱 [53] が挙げられる．これは，骨に付属する腱が石灰化を起こしたもので，骨化腱がみられる部分は骨の関節が柔軟に動かず，一本の梁のような役割をしたと考えれている．ヴェロキラプトルやデイノニクスなどの一部の鳥脚類では，尻尾の骨に骨化腱がみられるため，尻尾は硬く，根元だけを動かすことで動的な姿勢の保持に利用されたと考えられている．

また，尻尾の骨の形状が大きく異なるものも存在する．アンキロサウルスの尻尾

の先端はこん棒のようになっており，ハンマーのように尻尾を振ることで，敵から身を守ってきたと考えられている [53]．ステゴサウルスは，身体に放熱板と思われる骨が生えているだけでなく，尻尾先端に棘が生えている．これらの尻尾の特徴は，現存する生物には存在せず，その利用方法が敵からの防御法だけだったのか解明されていない．アンキロサウルスの尻尾の化石から，Arbour[54] は尻尾を振った際の威力がどのくらいかであったかを議論している．

1.4.2　ロボットへの応用

近年，移動ロボットの分野における研究では，尻尾がロボット本体に反力と反トルクを与える機構として注目されている．このような尻尾の利用法をロボットに初めて適用したのは，滝田ら [56] であると考えられる．それ以前にも尻尾を搭載したペット型ロボットの研究 [59][60] は存在するが，ロボットの振る舞いが人に与える影響を主とし，尻尾がロボット本体に与える影響は皆無である．2000 年に滝田らが，動的な運動のため，第 3 の脚として用いる等の複数の用途を目的とした恐竜型ロボットの提案を行った [56]-[58]．尻尾と首を 2 足の歩行に合わせて振ることで，安定に歩行と走行が可能であると示している．2008 年にヤモリの落下時における，尻尾を用いた姿勢の修正を模倣した Jusufi ら [61] の研究から始まり，尻尾に注目した研究が行われてきた．落下時の姿勢の修正に関する研究として，ネコが落下時に行う姿勢の修正 [62] が最も有名である．ネコは，重力以外の外力を得られないような状況であっても，体をひねることで，脚から着地できるよう姿勢を修正する [62]．この姿勢修正の原理やその応用について，多くの研究が行われている [63]-[68]．ネコが落下時に尻尾を用いて，姿勢の微調整を行っているとも提唱されている [69] が，これについては明らかにはなっていない [70]．Jusufi らは尻尾が姿勢修正に大きく影響を及ぼす生物として，爬虫類に注目し，それをロボットで模倣した [61][70]．

Jusufi らの提案の後，トカゲの跳躍時の模倣を車輪型ロボットに適用した研究が Chang-Siu，Libby らにより行われた [71][72]．これらの尻尾は，一本の剛体棒とその先端に位置する錘，根元のアクチュエータにより構成される．尻尾を振ることで，尻尾先端にかかる慣性力が，反力・反トルクとなってロボット本体に働く．浮遊時に

は，ロボット本体と尻尾をまとめた全体のシステムは 2 リンクの浮遊ロボットと表すことができ，Johnson らは主にアクチュエータの選定について議論している [73]．De ら [76] は，ホッピングロボットに尻尾を搭載し，安定した跳躍を実現している．他にも，尻尾を用いた浮遊時のロボットの姿勢制御を行う研究として文献 [74] や文献 [75] が挙げられる．

ロボットが接地している場合でも，尻尾からの反力・反トルクによりロボット本体は影響を受ける．チーターのように，ロボット本体を高速に回転させる研究が行われている．Pullin らは，8 足のロボット OctoRoACH に尻尾を搭載し，高速回転するための尻尾のコントローラを提案している [77]．ロボットが接地した状態での高速回転は，接地部と地面の摩擦が大きく関係する．OctoRoACH は脚の素材として摩擦係数の高いものが利用されている．Fisher らは車両型のロボットで高速回転を目指した [78]．高速回転するうえで，片側車輪が浮き，もう一方が設置している状態を振り子のモデルで表し，高速回転するためのコントローラを提案している．Briggs ら [79] はチーターそのものを模倣したロボット（MIT Cheetah）のための，ロボット本体に反力・反トルクを与える機構について議論し，尻尾がリアクションホイールよりも効率が良いと結論付けた．

これらの研究においても，尻尾先端に位置する錘の重量は，ロボット本体の 1/10 程度と設定している．一方で，Patel らはチーターの尻尾の重量はそれほど大きいものではないため，尻尾の重量による慣性力ではなく，尻尾にかかる空気抵抗がチーターの高速回転に影響を及ぼしているのではないかと示唆している [80]．実際に，跳躍を主な移動手段として用いているカンガルーネズミは，胴の倍もある尻尾の先端に房毛を有しており，空気抵抗を利用しすることで，空中で体の方向転換を行っている．

1.4.3　移動ロボットに尻尾を搭載した場合の利点

前々節と前節を合わせて考えることで，尻尾はロボットにおいて大きな利点があると考えられる．その例を図 1.6 に示している．例えば，ロボットがある出発点から，目的地にてタスクを実行する命令が与えられた状況を想定する．ロボットが移

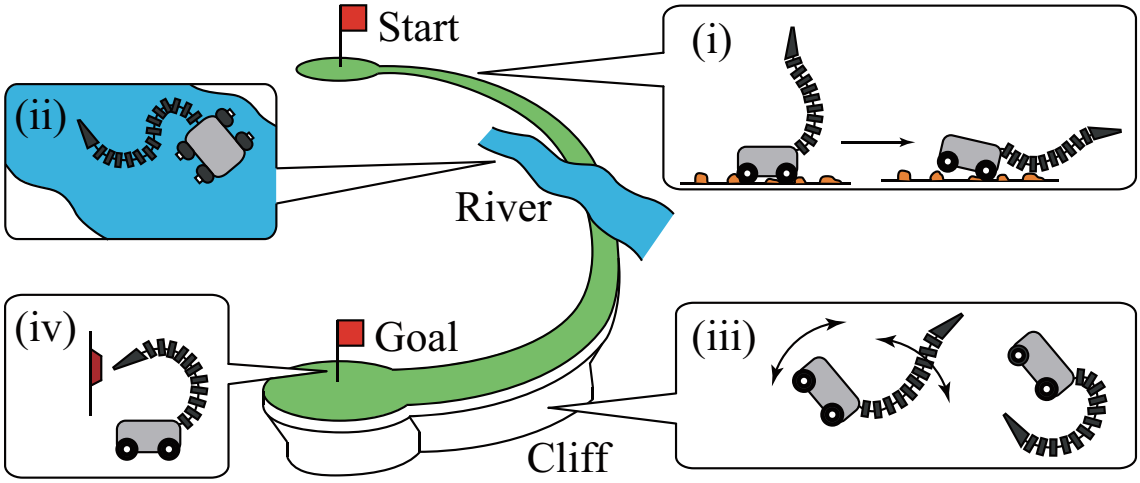


Fig. 1.6 多目的な尻尾の用途

動する経路は平坦とは限らない．斜面での移動では，尻尾によるバランス補償を行うことで，ロボット全体の重心を移動させ，転倒を防ぐことができる．不整地では，小さな石によって転倒や移動不能な状態に陥る可能性が存在する（図 1.6（i））．この時，尻尾による移動不能な状態からの脱却を試みることが可能である．また，ロボットが経路計画に用いた地図は完全であるとは限らない．地図上には存在しない小さな川が経路を塞いでいたり，経路が浸水している場合，川を横断する手段を持たないロボットは，再度経路計画を行い，迂回しなければならない．これは，目的地に到達するまでの時間ロスにつながる．魚を模したロボットの研究 [81]-[86] を参考とすることで，尻尾を用いて圧力差を生み出し，水中での移動も期待できる．さらに，図 1.6（iii）のように崖から落下してしまった場合でも，尻尾による姿勢修正や，尻尾を積極的に衝撃緩和に利用することで，ロボット本体の故障を防止することができる．尻尾の形状が変化しなければ，根元アクチュエータからのトルクを先端にて力に変換できて，スイッチを押すことや，物をひっかけるなどの簡単な仕事が可能であり，モノシリックアームのような使い方が考えられる（図 1.6（iv））．

尻尾が，これらの個々の状況に対処する機構として最適でないことは明白である．しかし，個々の状況に最適な機構を全てロボットに搭載することは，サイズ制限や重量制限，また価格面で不可能と言える．未知な環境にてロボットの移動能力を高

めるには，多くの状況に対処可能な機構を搭載することが一つの準最適な手段であると言え，本研究ではその機構として尻尾が挙げている．

1.4.4　提案する尻尾機構と跳躍動作制御手法

本稿では，多目的に利用可能な尻尾を想定する．Jusufi ら，Chang-Siu，Libby らが研究している剛体 1 リンクの尻尾は，用途が限定され，狭い場所での移動は難しいことが欠点として挙げられる．移動の妨げにならないことを考え，柔軟でありながら，尻尾を用いたタスク実行のため，剛性を変化可能な尻尾が好ましい．剛性を切り替えることが可能であるということは，尻尾の形状が保たれることと言い換えることができる．

ここで尻尾の先端に錘が搭載されており，尻尾の運動中に剛性を瞬間的に切り替えることが可能であるとする．尻尾の形状が変化可能であった状態から，変化不可能な状態に切り替わるため，尻尾の先端には速度方向とは逆向きに加速度が発生する．これは，先端の錘に慣性力となって働き，車両ロボットに反力・反トルクとなって伝達する．剛体 1 リンクの尻尾の先端は，根元アクチュエータの角速度に拘束されるが，柔軟尻尾では，尻尾長や尻尾の弾性，根元アクチュエータの入力等のパラメータを適切に設定することで，根元アクチュエータの角速度よりも大きな角速度となる時刻を作ることができる．この時刻で，瞬間的に剛性を変化させると，剛体 1 リンクの尻尾以上に大きな反力・反トルクを得ることができる．これを利用して，車両ロボットを宙に浮遊させることを考える．不整地という環境で問題となる凹凸を飛び越え，移動不能状態を回避するため，跳躍のためのモデルと動作制御手法を提案する．

本稿の尻尾機構と跳躍動作制御手法に関する結果は，既に文献 [3] に掲載されている．

第2章　未知環境での掃引動作制御に関する研究

2.1　緒言

掃引作業は二次元領域全面を覆い尽くす作業であり，清掃や塗装，物体探索など幅広い適用分野がある [94]．市販の家庭用掃除ロボットは，この作業を行う掃引ロボットの一例である．掃引ロボットにとって，掃引動作の効率の向上は重要であり，また，作業領域すべてを掃引したことを，ロボット自らが確認する完全性も信頼性の意味で非常に重要である．しかし，現行の家庭用掃除ロボットの多くは掃引済み領域に関する情報を持たないため，同じ領域を繰り返し掃引する可能性がある．このため掃引動作の効率は低く，完全性を確認できないため信頼性に問題がある．

本章では，家庭用掃除ロボットのような安価な掃引ロボットによる，掃引動作の効率向上と掃引完全性保障のためのオンライン地図生成と，これに基づく効率的動作生成を議論する．完全な掃引動作生成問題に関する多くの研究 [87] はセル分割法 [88] やその他の方法 [38][90] を用い，環境情報が既知である場合を取り扱っている．しかし，前もって詳細な地図を作成するコストが大きいこと，移動可能な家具などの存在により地図が変化すること，地図に基づいた正確な動作制御が困難なことから，完全な事前地図に基づく方法は実用的とは言えない．一方，地図生成を行いながら経路を計画する方法 [41][43] も提案されている．これらの方法では地図は事前に計画された掃引動作の間に生成され，最後にロボットは地図を用いて未掃引領域へ移動する．このような方法は環境変化に対処することはできるが，ロボットの姿勢（位置と角度）に関する正確な情報を前提とするため，実現のためのセンサシステムのコストは大きい．

また，部屋環境に設置したカメラを用いて掃引作業前に大まかな地図生成を行い，家具などによるオクルージョンが存在する場所は掃引作業中のロボット搭載センサを用いて地図を追加・修正しながら掃引動作を生成する手法も考えられる．しかし，掃引済み領域を推定するため，ロボットの動作を正確に測定する必要があることや，ロボット外のカメラで取得した部屋環境情報とロボットのセンサ情報を統合するシステムが必要なことなど，ハード・ソフトともそのコストは大きくなる．

本研究ではシステム簡素化のため，ロボット搭載のローカルセンサ情報により地図情報を得る方法を採用する．ここでのローカルセンサとは検出範囲の狭い赤外線センサや接触センサだけでなく，ロボット搭載の RFID タグリーダやカメラ，レーザーレンジファインダ [92] 等も含まれ，局所情報を得るものを意味することとする．

家庭用掃除ロボットのような低コストの掃引ロボットは，いくつかの基本動作からなる行動ベースの動作 [31] を採用している．行動ベースの動作は環境モデルを持たないため，環境変化や制御誤差に対してロバストであるものの，一般に掃引動作の効率は低く，掃引動作完了を確認することはできない．

掃引完了の確認を行い，かつ，効率的動作を生成しようとする場合は，掃引済み領域を含む地図情報が必要である．上述のように，ここではロボットが持つローカルセンサ情報により，オンライン地図生成をおこなう．オンライン地図生成を漸進的に行っている場合は，以下の問題を生じる．すなわち，地図生成当初では地図はほとんどできておらず，極めて不完全な地図のため，大域的かつ効率的動作生成は困難であること．また，ほぼ完全な地図が生成できたときには，ほとんどの領域は掃引済みとなっており，その時点での大域的動作は，掃引作業の点からすでに意味を失っていることである．

本研究では，上記の問題に対処するため，漸進的に生成される地図情報を有効に用いながら，掃引効率ができるだけ高い動作を生成する手法を提案する．提案する方法は，当初は局所情報に基づく動作生成を主としながら，地図生成中盤から後半は作成されつつある地図情報に基づいて大域的で効率的動作を生成する手法である．特に地図生成後半の状況では，未掃引領域の塊（以下，未掃引コロニーと呼ぶ）が掃引領域に散在する状況となる．既存の掃引ロボットの研究では，掃引ロボットの移

動誤差やセンサー誤差を考慮していないため，掃引途中に発生する未掃引領域の塊に対する効率的な掃引計画問題はほとんど検討されてこなかった．本研究では，このような，複数の未掃引コロニーが散在する状況に対して掃引計画を行う手法も提案する．

本章における提案の中心は主に次の 2 点である．1 つ目は漸進的地図生成に伴い，局所的な動作計画から大域的な動作計画に自動的に切り替わる動作戦略．2 つ目は，センサ誤差や位置誤差が大きな場合に，掃引作業後半で現れる未掃引領域が複数の塊に対する掃引方法である．

本章では，まず掃引戦略と掃引効率に関する議論を行い，3 節にて提案する掃引アルゴリズムを提案する．次に，提案するアルゴリズムの有効性を示すため，掃除ロボットを例として取り上げる．4 節で想定する掃除ロボットのシステムを述べ，5 節で提案アルゴリズムを掃除ロボットに適用した場合の例を述べる．6 節では，ロボットのセンサ誤差が無視できない場合の掃引作業で発生する，新たな問題点について述べ，これを解決するための手法について提案する．4 節から 6 節までのアルゴリズムを用いて，掃除ロボットのシミュレーションと実験を行うことで，提案するアルゴリズムが Behavior-based アルゴリズムを用いたロボットに比べ，有効であることを示す．8 節では，本提案アルゴリズムに関する考察を行う．

2.2　掃引戦略と掃引効率

効率的かつ確実な掃引ロボット動作制御を検討するため，掃引戦略と掃引効率の関係を考える．ここでは掃引問題を以下のように設定する．

1) ロボットは 2 輪独立駆動型であり左右輪ともに最大速度は ω，車輪半径は r_w とする．

2) 動作環境は壁面に囲まれたフラットな床面であり掃引可能面積は S.

3) ロボットの掃引幅は w.

このとき単純化のため，直進や回転に関わらずロボットは常に最大速度 $v = r_w\omega$ で走行し，旋回動作や停止などにおける時間ロスは考慮しないとし，また掃引に全く重複がなければ，掃引完了時間は図 2.1 に示す $T_0 = \frac{S}{vw}$ で見積もることができる．

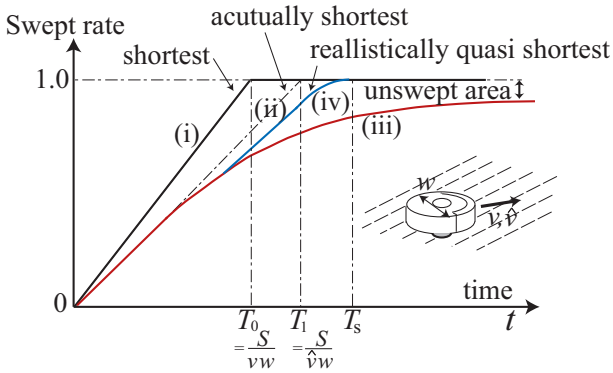


Fig. 2.1 When we assume that a robot can sweep a room without any overlap and the robot always run at the maximum speed, the swept rate expected to be the black line (i) which means ideal shortest time motion. The dashed line (ii) shows an actually shortest motion which includes realistic overlap and decreased velocity at turnings. However practical motion of the sweeping robot corresponds to red curve (iii), because overlapped sweep motion increases as time advances. The purpose of this paper is to realize more efficient motion than the actual motion (iii) which is shown by the blue line (iv) as a realistically quasi shortest time motion.

このとき縦軸を掃引率，横軸を時間とすると，掃引速度は図 2.1 のグラフ (i) の傾きとなる．これは効率の理論限界である．ここでいう掃引率 r とは，既掃引面積 S_{swept} を全体の面積 S_{all} を用いて，$r = S_{swept}/S_{all}$ で表し，掃引した割合を意味する．

一方，実際の掃引作業では掃引動作の不正確さのため，多少の掃引領域の重なりが必要であり，また，壁での一端停止や旋回動作での速度低下などから，部屋形状の複雑さが同程度であれば，ほぼ経路長に比例して掃引領域重複や時間ロスが生じると考えられる．このときの移動速度を平均として $\hat{v}$ と見積もるなら，図 2.1 の (ii) のグラフがこのときの掃引速度となる．これは同じ部分を掃引しない理想的な軌道であり，実質的な最適軌道と考えられる．もし，自由空間が凸形状であり地図が与えられており，ロボットは計画経路を完全に追従制御できるなら，水平パラレル動作 [89] や輪郭パラレル動作 [95] によりこの動作を実現できる．

しかしながら，実際の動作環境は凹形状で複雑なため，最適軌道を見つけることは困難である．また，事前地図情報がない場合は少なくとも最初は，計画的動作を

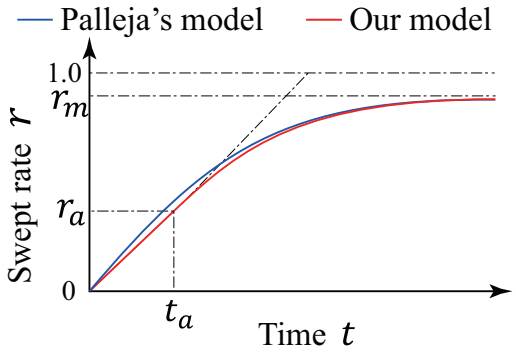


Fig. 2.2 Complete Coverage Model

用いることができず，反射的動作により掃引制御を行うことになる．市販の掃除ロボットは確率要素を含む数種類の反射動作を用いて掃引動作を実現し，そこそこの効率的な動作を行っている．このような動作の掃引率時間推移を見ると，典型的には図 2.1 の (iii) のようになる．Palleja ら [31] はこの反射的動作ロボットの掃引率はほぼ指数収束すると指摘しているが，著者らのシミュレーションや実験では，はじめはほぼ直線に近い変化，そののちは指数収束に近い変化をする [47]．また，これら反射行動型ロボットでは一般に実用的有限時間で掃引率を 1 に収束させることは困難である．これは動作初期においては重複が少ないが，後半では重複が増加し，未掃引地図情報を参照しないために完全な掃引が難しいことを示している．

我々のモデルは，

$$r(t) = \begin{cases} \dfrac{r_a}{t_a}\, t & (t \le t_a) \\ \\ r_m - (r_m - r_a)e^{-\kappa(t-t_a)} & (t > t_a) \end{cases} \tag{2.1}$$

で表される．ただし，

$$\kappa = \frac{r_a}{(r_m - r_a)t_a} \tag{2.2}$$

上式は $t = t_a$ 時における微分の連続性も考慮されている．

$$\dot{r}(t) = \begin{cases} \dfrac{r_a}{t_a} & (t \le t_a) \\ \\ (r_m - r_a)\kappa e^{-\kappa(t-t_a)} & (t > t_a) \end{cases} \tag{2.3}$$

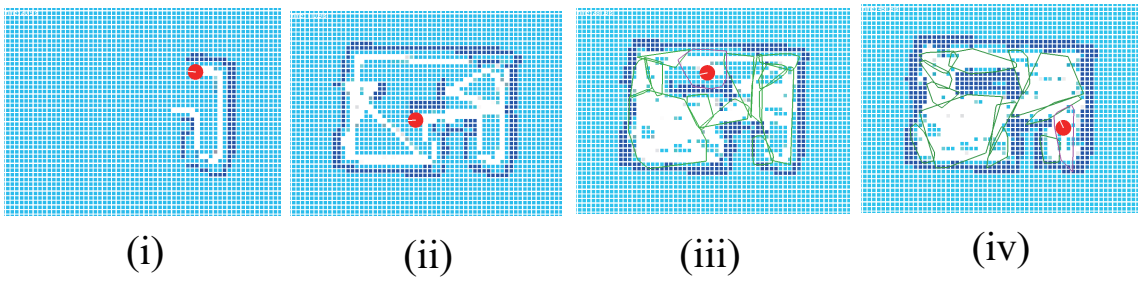


Fig. 2.3 The snapshot of the gradually building map.

当然ながら，生成地図を参照するロボットの場合，生成地図を参照しない場合に比べ，t_a は大きくなる．

本研究においては，最初は反射的行動をとりながらも，実時間で漸進的に生成されつつある大域的情報を用い，計画行動により，できるだけ効率的な掃引制御を行う．これらの動作選択をどのように実現するかも重要である．これにより有限時間での掃引完了保証をめざす．その結果，図 2.1 で (iv) に近い掃引速度となることが期待される．

2.3　提案する掃引動作制御アルゴリズム

ロボットの自己位置推定誤差が存在するような条件は，実環境では容易に考えられる．自己位置推定誤差が存在する場合，ロボットは計画した経路を移動できたかを判断することができない．また，作成する地図は曖昧さを含むため，複数回ロボットが通過することで確定する．掃引開始時に地図を全く持たないとしてリアルタイムで地図生成をおこなう時，図 2.3 の (i) から (iv) の順に遷移していく．掃引開始時には全て未確認領域であり，掃引完了時に完全な部屋形状がわかることになる．図 2.3 の（i）や (ii) の段階では，地図から環境全体を把握することができず，従来の地図全体を利用した効率的な掃引経路生成手法が適用できない．しかし，図 2.3（iii）では，地図が概ね確定しており，未掃引領域の判定が容易である．我々は図 2.3 の (iii) と (iv) のような状況で効率的な動作計画をおこなわなければならない．

掃引の終盤では，地図の信頼性が高いため，地図全体の情報を用いて，未掃引の

領域までの経路を計画し，掃引を行うことが可能である．一方，ロボットの周辺に未掃引領域が多く存在する場合，特に掃引初期では，地図情報の獲得が不十分なため，複雑な経路生成は障害物の発見や地図の大きな修正により，計画経路が無駄となってしまう．地図全体を用いた計画は，計算コストが大きいことから，このような場合は，簡単な動作制御手法が好ましい．しかし，地図情報を全く用いずに移動を行うと，既掃引領域上をロボットが移動してしまうため，地図全体の情報を用いるのではなく，ロボットの周辺の地図情報を用いた動作制御手法を採用する．

このとき，地図全体の情報を用いた動作制御手法と，ロボット周辺の地図情報だけを用いた動作制御手法の，2 つの手法をどのように切り替えるかという点が問題となってくる．図 2.4 に 2 つの手法の切り替え例を示す．横軸が掃引終了時刻を 1 として無次元化された時間，縦軸が動作制御を選択する割合を表している．図 2.4 の緑で示した Local Reference-based と記載された部分がロボット周辺の地図情報だけを用いた動作制御手法を表し，赤で示した Global Reference-based と記載された部分が地図全体の情報を用いた動作制御手法を表している．図 2.4 は，時間ごとの動作制御の選択割合を表している．図 2.4(a) では，ある時刻 t_1 を設定し，掃引開始から t_1 までを Local Reference-based 動作制御手法，t_1 から掃引終了までを Global Reference-based 動作制御手法を選択することを意味している．しかし，未掃引領域での掃引を考えているため，ロボットは環境の大きさや掃引率を地図情報から判断することができない．そのため，図 2.4(a) のようにある時刻 t_1 から 2 つの手法を切り替える手法は適さない．図 2.4(b) では，Local Reference-base 手法から Global Reference-based 手法へ，掃引が進むにつれ徐々に切り替える手法を意味している．掃引の終盤になるにつれ，地図の完成度が高くなることから，図 2.4(b) の手法は望ましいと言えるが，t_2 や t_3 をあらかじめ設定することは図 2.4(a) の場合同様に適切ではない．環境の大きさや形状に合わせて，t_2 や t_3 のようなパラメータが自動的に変化することが望ましいと言える．

さらに，Stachniss ら [48] が提案したように，掃引中にロボットの位置に関する不確かさを率先して改善する動作制御が必要になってくる．ロボットは自己位置の不確かさを監視し，掃引を継続するか，不確かさの改善を行うかを選択することが重

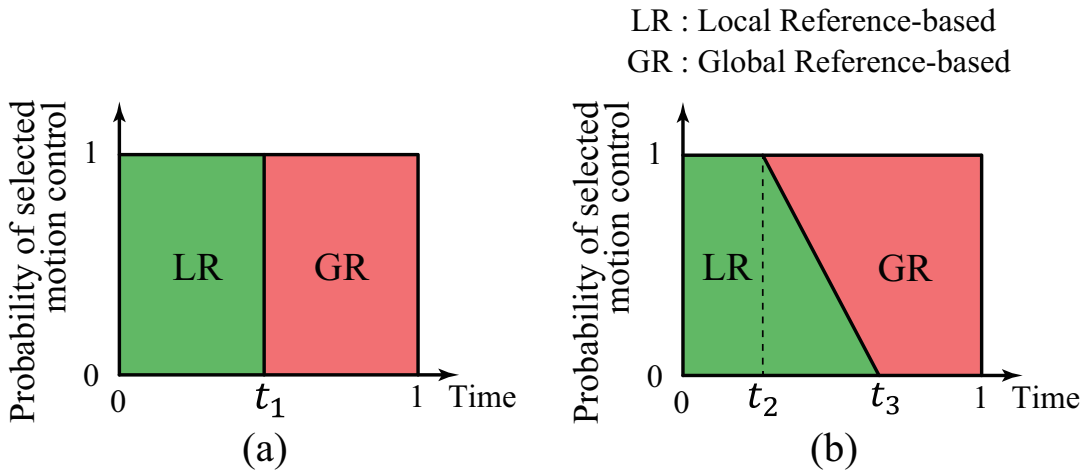


Fig. 2.4 These figures show the examples in which the robot selects either Local Reference-based motion control or Global Reference-based one.

要である．これを Landmark Search Control と呼ぶこととする．ロボットの自己位置の不確かさは，誤差の大きなセンサ情報の取得により大きくなる．さらに，ロボットが Landmark Search Control を選択したとしても，改善までには時間を要し，この時間は一定ではない．そのため，Landmark Search Control が選択される割合は，掃引時間に依存しないと考えられる．図 2.4(b) に Landmark Search Control を時間に関わらず，一定の割合で選択されるようなアルゴリズムが，効率の良い掃引アルゴリズムであると考えられる．

提案する掃引アルゴリズムを図 2.5（a）に示している．提案するアルゴリズムは，3 つの動作制御手法と 2 つの条件分岐により構成される．3 つの動作制御をそれぞれ，ロボット周辺の地図情報に基づいた動作制御手法 Local Reference-based Control と部屋全体の地図情報に基づいて動作計画を行う手法 Global Reference-based Control，ロボットの自己位置を修正するためのランドマークを探す動作 Landmark Search Control と呼ぶ．以降，Local Reference Contorl を LR，Global Reference Control を GR，Landmark Search Control を LS と省略する．2 つの条件分岐は，Check1 が「ロボットの周辺に未掃引領域が存在するか」を判断する条件，Check2 が「自己位置の不確かさを表す値が閾値より大きいか」を判断する条件を意味している．これ

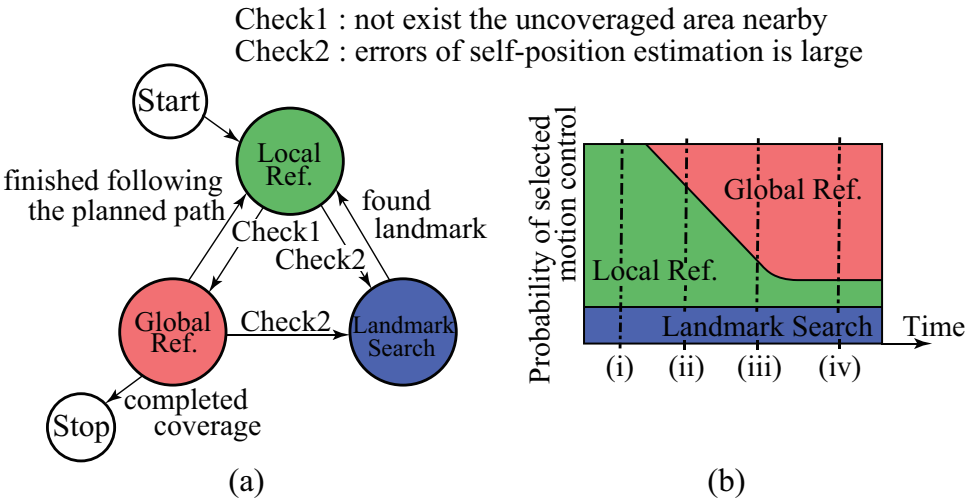


Fig. 2.5 Complete Coverage Strategy of GBM-based Algorithm

らの動作制御手法と条件分岐を掃引ロボットのシステムに合わせて，適切に設計することで，効率の良い掃引を行うことができる．

掃引ロボットは，LR を基本の動作制御として，Check1 や Check2 の条件が満たされたとき，他の動作制御手法を選択する．ロボット周辺の地図情報を参照し，周辺の未掃引領域の量が多い場合には LR を，少ないまたは存在しない場合には，Check1 により，GR を選択する．GR により計画された経路の追従が終了，中断されたとき，ロボットは LR を選択して掃引を行う．掃引ロボットは，自己位置の不確かさが大きいと推測された場合，Check2 により，LS を選択する．自己位置の不確かさを改善できたとき，ロボットは LR を選択する．

図 2.5（b）に，動作制御の概念図を示している．図 2.5（b）の (i) から (iv) は図 2.3 の (i) から (iv) に対応する．提案アルゴリズムの Check1 を設定することで，図 2.4(b) の t_2 や t_3 を設定せずとも，LR と GR は掃引が進むにつれ徐々に切り替えられ，t_2 や t_3 にあたるパラメータは環境の大きさや形状によって自動的に変化するという部分が，本研究の大きなポイントである．本提案アルゴリズムは，Check1 を基に漸進的生成地図を参考にしながら，掃引のための 2 つの動作制御を切り替えるため，Gradually Building Map-based（GBM-based）アルゴリズムと呼ぶ．掃引開始

時には，周辺に未掃引領域が多く存在するため，基本的には LR が選択される（図 2.5（b）の（i））．時刻が経過すると，ロボット周辺に未掃引領域が存在する確率が減少するため，GR の選択割合が徐々に高くなる（図 2.5（b）の（ii）や（iii））．最終的に周辺には未掃引領域はほとんどなくなるため，GR の選択割合がほとんどを占める（図 2.5（b）の（iv））．つまり，ロボットは環境全体の大きさを把握していなくても，確率的に時刻とともに局所的な動作制御から大域的な動作制御に切り替わる．また，ロボットの自己位置の不確かさの上昇は時間に依存しないと考えられるため，時間に関わらず一定の割合で LS が選択される．

2.4　想定する掃除ロボットとそのシステム

本節では，提案するアルゴリズムを掃除ロボットに適用するため，想定する掃除ロボットについて述べる．

想定する掃除ロボットの図を 2.6 に示している．本研究では，一般的な家庭内掃除ロボットを想定し，図 2.6 に示すシステムとする．ロボットはセンサとしてタイヤの回転を計測するためのエンコーダ，衝突を検知するタッチセンサ，近距離の壁を計測するための赤外線距離センサを左右に 1 つずつ有している．2 つのタッチセンサを同時に検知した場合に，ロボットは正面の壁に衝突したとみなす．赤外線センサは図 2.6 のように，左右ともに水平方向から 55 ℃の角度に取りつけられている．

ロボットが地図を用いて掃引を行う際，過去にその場所を掃引して作成した地図を利用することが考えられる．しかし，たとえば椅子などの障害物は簡単に移動可能であり，そのことは，作成地図の大域的構造を大きく変化させる可能性があるため，そのような動的障害物がある場合は，過去の地図は信頼できない．本例題では，このことを考慮し，掃引開始時には新しく地図を作り直し，過去の地図情報は用いないとする．

ロボットが部屋形状情報を得る方法として，ランドマークを用いる手法がある．本例題では，家庭用の掃除ロボットも対象としているため，安価で簡便な RFID タグをランドマークとして用いる．RFID タグの設置方法として，多くのタグを床面に

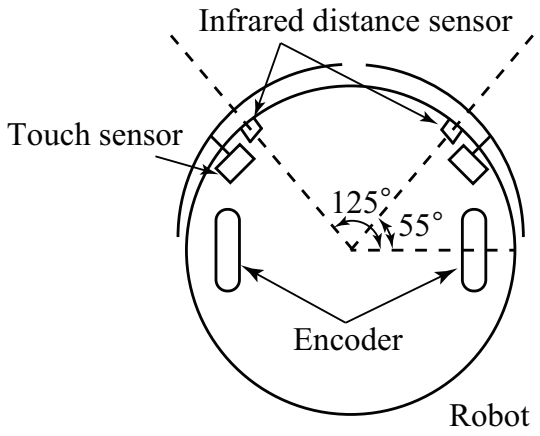


Fig. 2.6 The system of a sweeping robot for the simulation to confirm the validity of proposed algorithm

一様に取り付ける方法 [93] があるが，本例題ではより簡便な方法として，比較的少数のタグを部屋の周囲壁に取り付けることとする．なお，ここで提案する掃引制御手法は，RFID タグの利用に限定しているのではなく，1 章でも述べたようにオンラインでロボットの位置姿勢を推定可能なものであれば，他のロボット搭載センサ等でもかまわない．

RFID タグには通信距離が短距離のものから長距離のものが存在する．長距離や中距離の RFID タグは，重複した複数の RFID タグを検出することでロボットの位置を推定する方法である．しかし，実際は RFID タグの検出範囲は円でないため，推定精度を高めることは簡単ではない．一方，短距離 RFID タグは検出範囲が狭いため検出位置の推定精度が高いが，検出すること自体が難しくなる．ここでは，ロボットの壁面追従動作により頻繁に検出できるよう取り付けることで，この欠点は補うことができる．そのため，本例題では短距離の RFID タグを用いる．本章ではまず，短距離 RFID タグ設置環境における軌道推定方法について述べ，次に生成する地図について述べる．

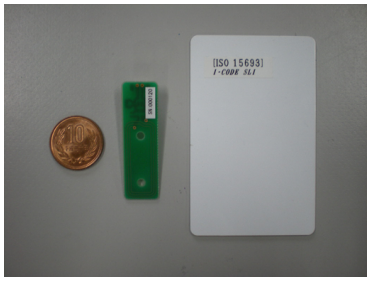


Fig. 2.7 RFID antenna and tag using in the robot.

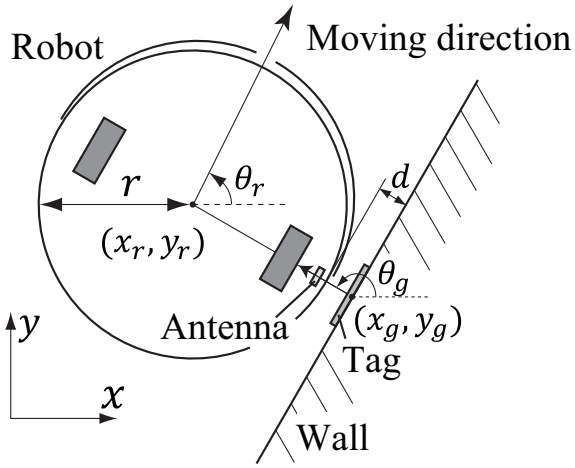


Fig. 2.8 Geometric relationship between robot and RFID tag.

2.4.1　RFID タグ観測モデル

RFID タグには，タグを張り付ける位置の座標とタグの姿勢が書き込まれており，ロボットはこれを利用できる．タグのリーダアンテナは，ロボットが進行方向右側に壁を追従することを考慮して，図 2.8 に示すように円形のロボットの右側面に取り付ける．また，ロボットの姿勢 x_r, y_r, θ_r と検出されたタグの位置と角度 x_g, y_g, θ_g の幾何関係を図 2.8 に示す．ここで，r はロボットの半径を表し，d はロボットと壁の距離を表す．通信距離が短いため，d は小さな定数とみなす．図 2.8 の幾何関係は以下の式で表される．

$$\begin{pmatrix} x_r \\ y_r \\ \theta_r \end{pmatrix} = \begin{pmatrix} x_g \\ y_g \\ \theta_g \end{pmatrix} + \begin{pmatrix} (r+d)\cos\theta_g \\ (r+d)\sin\theta_g \\ -\pi/2 \end{pmatrix} \tag{2.4}$$

式（2.4）において，ロボットの姿勢 x_r, y_r, θ_r は実際には RFID の通信範囲内で小さい誤差を含む．その誤差を平均 0 のガウス白色雑音と近似し，ロボット掃引軌道を以下で述べる確率モデルで表現する．

2.4.2　拡張カルマンフィルタによる掃引軌道推定

拡張カルマンフィルタ (EKF)[18] を用いて自己位置・掃引軌道推定を行う．EKF により推定したロボットの姿勢推定値と誤差を表す共分散からセンサ情報を組み合わせて，既掃引領域と障害物・壁領域である確率値を算出する．

本例題ではロボットの並進速度 v_t と回転速度 ω_t からなるロボットのオドメトリを制御入力とみなす [16]．状態方程式はロボットの姿勢 x_t, y_t, θ_t を用いて以下のように表される．

$$\begin{pmatrix} x_{t+1} \\ y_{t+1} \\ \theta_{t+1} \end{pmatrix} = \begin{pmatrix} x_t \\ y_t \\ \theta_t \end{pmatrix} + \begin{pmatrix} (v_t + w_{vt}) \cos\theta_t \Delta t \\ (v_t + w_{vt}) \sin\theta_t \Delta t \\ (\omega_t + w_{\omega t}) \Delta t \end{pmatrix} \tag{2.5}$$

ここで $w_{vt}, w_{\omega t}$ は v_t, ω_t の雑音を表し，Δt は計算時間間隔を表す．観測方程式は観測モデルを用いて以下のように表される．

$$\begin{pmatrix} x_{rt} \\ y_{rt} \\ \theta_{rt} \end{pmatrix} = \begin{pmatrix} x_t \\ y_t \\ \theta_t \end{pmatrix} + \begin{pmatrix} v_{xt} \\ v_{yt} \\ v_{\theta t} \end{pmatrix} \tag{2.6}$$

ここで，$v_{xt}, v_{yt}, v_{\theta t}$ は x_t, y_t, θ_t に加わる雑音である．式（2.5）と式（2.6）より，以下の非線形システムモデルを得る．

$$\boldsymbol{x}_{t+1} = \boldsymbol{f}_t(\boldsymbol{x}_t, \boldsymbol{u}_t + \boldsymbol{c}_t) \tag{2.7}$$

$$\boldsymbol{z}_t = \boldsymbol{h}_t(\boldsymbol{x}_t) + \boldsymbol{e}_t \tag{2.8}$$

ここで，状態 $\boldsymbol{x}_t = (x_t, y_t, \theta_t)^T$，制御入力 $\boldsymbol{u}_t = (v_t, \omega_t)^T$，観測 $\boldsymbol{z}_t = (x_{rt}, y_{rt}, \theta_{rt})^T$ である．制御雑音 $\boldsymbol{c}_t = (c_{vt}, c_{\omega t})^T$ と観測雑音 $\boldsymbol{e}_t = (e_{xt}, e_{yt}, e_{\theta t})^T$ は共分散 $\boldsymbol{Q}_t, \boldsymbol{R}_t$ を持つ平均 0 のガウス白色雑音とする．

状態の推定 $\hat{\boldsymbol{x}}_{t/t}$ とその共分散 $\boldsymbol{P}_{t/t}$ を計算する更新式は以下のように記述される [18][16]．

1. 時間更新

$$\hat{\boldsymbol{x}}_{t+1/t} = \boldsymbol{f}_t(\hat{\boldsymbol{x}}_{t/t}, \boldsymbol{u}_t) \tag{2.9}$$

$$\boldsymbol{P}_{t+1/t} = \hat{\boldsymbol{F}}_t \boldsymbol{P}_{t/t} \hat{\boldsymbol{F}}_t^T + \hat{\boldsymbol{G}}_t \boldsymbol{Q}_t \hat{\boldsymbol{G}}_t^T \tag{2.10}$$

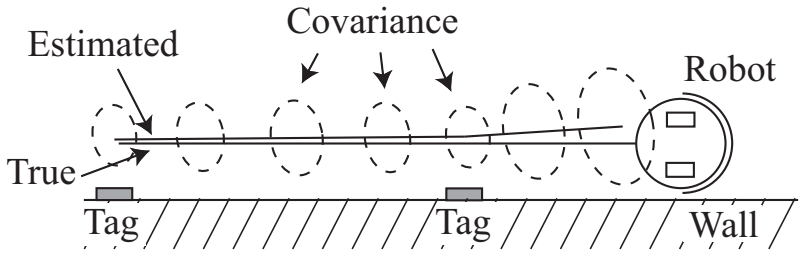


Fig. 2.9 The robot estimates its pose and position using extended Kalman filter (EKF). The estimated pose and position are updated by odometry information at each time step, and their error increase as time advances. When the robot reads new information from a tag, the robot's estimated pose and position are updated more accurately. Then, the estimated trajectory of the robot are modified using fixed-Interval smoothing.

2. 観測更新

$$\hat{\boldsymbol{x}}_{t/t} = \hat{\boldsymbol{x}}_{t/t-1} + \boldsymbol{K}_t[\boldsymbol{z}_t - \boldsymbol{h}_t(\hat{\boldsymbol{x}}_{t/t-1})] \tag{2.11}$$

$$\boldsymbol{K}_t = \boldsymbol{P}_{t/t-1}\hat{\boldsymbol{H}}_t^T[\hat{\boldsymbol{H}}_t\boldsymbol{P}_{t/t-1}\hat{\boldsymbol{H}}_t^T + \boldsymbol{R}_t]^{-1} \tag{2.12}$$

$$\boldsymbol{P}_{t/t} = (\boldsymbol{I} - \boldsymbol{K}_t\boldsymbol{H}_t)\boldsymbol{P}_{t/t-1} \tag{2.13}$$

ここでは，以下の線形近似を用いる．

$$\begin{cases} \hat{\boldsymbol{F}}_t = \left(\frac{\partial \boldsymbol{f}_t}{\partial \boldsymbol{x}_t}\right)_{x=\hat{x}_{t/t}} \\ \hat{\boldsymbol{G}}_t = \left(\frac{\partial \boldsymbol{f}_t}{\partial \boldsymbol{u}_t}\right)_{x=\hat{x}_{t/t}} \\ \hat{\boldsymbol{H}}_t = \left(\frac{\partial \boldsymbol{h}_t}{\partial \boldsymbol{x}_t}\right)_{x=\hat{x}_{t/t-1}} \end{cases} \tag{2.14}$$

ロボットの姿勢はステップごとにオドメトリ情報により更新され（時間更新式），その不確かさは増加する．ロボットがタグを検出するとロボットの姿勢は修正され（観測更新式），同時に不確かさは減少する．

タグ検出時，ロボットの姿勢は EKF により修正されるが，タグ検出の直前の軌道は不確かなままであるため，タグ検出後も地図はタグ検出前と変わらず，既掃引領域の判定ができない．本例題では固定区間スムージング [18] により，過去の軌道の

修正を行う（図 2.9）．固定区間スムージングはある区間 $[0, N]$ のすべてのロボットの位置姿勢の最小分散推定量とその共分散を出力する．オンラインで処理を行うため，本例題では現在のタグ検出時刻と一つ前のタグ検出時刻の間の区間のみ軌道の修正を行う．

固定区間スムージングは以下の二つのステップからなる [18]．

1. 区間 $[0, N]$ において EKF により推定 $\hat{\boldsymbol{x}}_{t/t-1}, \hat{\boldsymbol{x}}_{t/t}$ と共分散 $\boldsymbol{P}_{t/t-1}, \boldsymbol{P}_{t/t}$ を計算し記憶する．

2. 時刻 $t = N-1$ から $t = 0$ まで逆向きに $\hat{\boldsymbol{x}}_{t/N}$ と $\boldsymbol{P}_{t/N}$ を計算する．

$t = N-1$ から $t = 0$ へのロボットの姿勢と共分散の修正式 [18] は以下のようになる．

$$\hat{\boldsymbol{x}}_{t/N} = \hat{\boldsymbol{x}}_{t/t} + \boldsymbol{C}_t[\hat{\boldsymbol{x}}_{t+1/N} - \hat{\boldsymbol{x}}_{t+1/t}] \tag{2.15}$$

$$\boldsymbol{C}_t = \boldsymbol{P}_{t/t}\hat{\boldsymbol{F}}_t^T(\boldsymbol{P}_{t+1/t})^{-1} \tag{2.16}$$

$$\boldsymbol{P}_{t/N} = \boldsymbol{P}_{t/t} + \boldsymbol{C}_t[\boldsymbol{P}_{t+1/N} - \boldsymbol{P}_{t+1/t}]\boldsymbol{C}_t^T \tag{2.17}$$

上式を用い，それぞれの時刻におけるロボットの姿勢と共分散はより正確に推定される．タグの検出ごとに以上の処理を繰り返し行い，ロボットの軌道をオンラインで推定する．

2.4.3　生成地図と地図の更新

推定したロボットの姿勢とその共分散を用いて，作業環境の既掃引領域と壁形状を推定する．壁形状の推定により到達可能な未掃引領域と到達不可能な未掃引領域を区別でき，既掃引領域の推定により掃引動作の完全性を確認できる．想定するシステムでは，既掃引領域と壁形状両方の推定に一つの占有格子地図 [12] を用いる．占有格子地図は環境を分割するそれぞれのセルに対して物体の占有確率 p を定義する．ロボットが底面全面で掃引すると想定し，既掃引領域と自由領域，未掃引領域と未知領域を同一のものとする．地図 $\boldsymbol{m}$ の更新時に，$\boldsymbol{m}$ の 1 つのセル m_i における占有確率を独立に計算する．セル m_i の占有確率 $p(m_i)$ の値が 1 のとき m_i は障害物・

壁，$p(m_i)$ が 0 のとき既掃引（自由空間），$p(m_i)$ が 0.5 のとき未掃引（不明な空間）を意味する．軌道推定により求めた位置および掃引ロボットが持つセンサ情報に基づき占有格子地図を更新する．

掃引ロボットは，タッチセンサおよび壁面までの距離を計測する距離センサを持つものとする．このタッチセンサは前方左右の接触方向を検出できるものとする．すなわち接触方向に関して 90 度の範囲の円弧を考え，その円弧が存在するセルには占有（壁）が観測される．本例題ではロボットの底面全面で掃引を行うと設定し，セルの中心とロボット中心の距離が半径以内であればフリー（掃引）とみなす．EKF により求めた姿勢推定値と共分散に，その時刻でのセンサ情報を組み合わせることで，既掃引領域と障害物・壁領域をそれぞれ確率に変換する [47]．地図（掃引領域と壁形状）作成処理を軌道推定と同時に行う．通常は拡張カルマンフィルタの推定に基づき地図を更新する．タグの観測により固定区間スムージングの修正が行われた時点で軌道を修正した地図に更新する．以上の処理を繰り返すことにより，軌道推定および地図作成を実時間で実行する．

2.5　掃除ロボットを想定した掃引動作制御例

本節では，前節での説明した掃除ロボットシステムに提案する掃引アルゴリズムを適用した例について述べる．Local Reference-based 手法と Global Reference-based 手法，Landmark Search 手法と 2 つの遷移する条件について説明する．

2.5.1　掃引動作戦略

掃除ロボットに適用した掃引動作戦略の概念図を図 2.11 に，掃引動作アルゴリズム概略を図 2.10 に示す．提案する掃引動作は壁追従動作，局所情報を用いた動作制御，大域情報を用いた動作制御の 3 つの動作制御から構成される．動作制御の切り替えは壁衝突時に自動的に行われる．

壁衝突時に，自己位置推定誤差が大きい場合，計画動作により掃引制御しても正確な掃引は期待できない．従って，この場合は壁追従動作を選択することで，自己

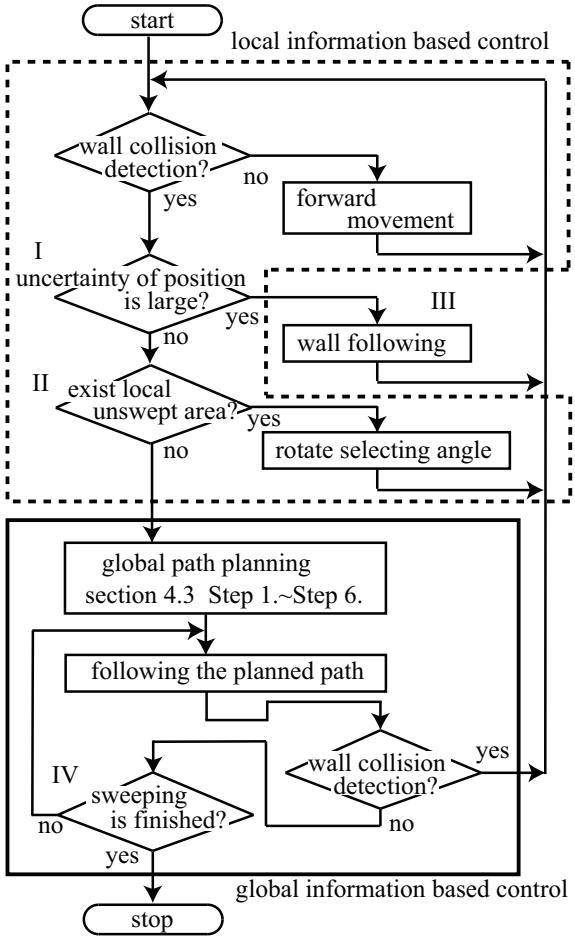


Fig. 2.10 The algorithm of the sweeping motion control is divided into three sub–controls. The first one is the local information based control which is shown inside the dotted line. When the robot collides a wall, it causes a transition to another state (I). If the uncertainly of the robot's position is large in the state (I), transition to the wall following (state III) is caused. This wall following control (second sub–control) is important to find RFID tags. If the unswept area is not detected in the neighborhood of the current position when the robot collides the wall (state II), the control is changed to the global information based control (third sub-control) which is shown in the solid line box.

位置修正を図る（図 2.10 の I）．本稿では，自己位置推定誤差はロボットの自己位置誤差を表す共分散行列の最大固有値を用いて評価し，設定した値より大きい場合は

壁追従動作を選択する．自己位置推定誤差が小さい場合，ロボットは推定位置周辺の地図情報を確認する．このとき，地図周辺にある程度掃引可能な面積が存在する場合，局所情報を用いた動作制御を，周辺に適当な未掃引領域が少ない場合は，大域情報を用いた動作制御を自動選択する（図 2.10 の II）．本稿では，掃引可能面積は後述するエントロピーを用いて算出する．

図 2.11 は横軸を掃引率 $r = [0, 1]$ とし，掃引が開始されてからロボットが選択する動作制御の割合を概念的に表したものである．(a) や (b) のように掃引開始直後では，地図全体の完成度は低いため，ロボットはほとんど局所情報を用いた動作制御を選択することになる．(c) のようにある程度掃引が進むと，ロボットの周辺に未掃引領域があまり存在しない状況が発生する．そのため大域情報を用いる動作制御を選択する割合が増えてくる．掃引終盤の (d) では，大域情報を用いる動作制御を選択する場合がほとんどとなる．正確な地図を生成するためには，自己位置推定精度を高めるため，壁を追従することが必要である．そのために，掃引時間にかかわらず一定の割合で壁追従動作を選択する（図 2.10 の III）．

2.5.2　Local Reference-based 動作制御

図 2.10 の破線枠で示す局所情報を用いた動作制御を説明する．この動作においても，掃引動作の効率を向上させるためには，ロボットは地図を用いてできるだけ多くの未掃引（未知）の経路を通過することが望ましい．そこでまず，各格子セルに対して，その格子セルの観測時に得られる情報量を意味するエントロピーを計算する．未知の格子セルを最も高い値として設定するエントロピーは，格子セル m_i の占有確率 $p(m_i)$ を用いて

$$H_p(m_i) = -p(m_i)\log(p(m_i)) - (1 - p(m_i))\log(1 - p(m_i)) \tag{2.18}$$

により求められる [16]．多くの高いエントロピーのセル，すなわち未掃引の度合いの高いセルを通過する経路が効率的であると考えられる．

すべての経路に関して取得可能なエントロピーの和を求め，その和が最大となる経路を選択することが望ましい．しかし，計画可能な経路は無限に存在するため，そ

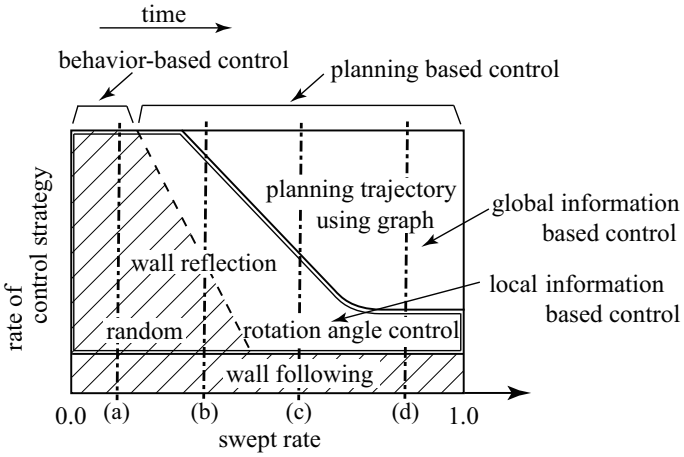


Fig. 2.11 This figure illustrates the conceptual diagram of the sweeping control strategy. There are three sub-control which are the wall following, the local information based control and the global information based control. When the robot collides the wall, the control may be changed depending on situations. The robot's control strategy　gradually changes from local strategy to global strategy as time advances without any knowledge of the room and obstacles, by using the proposed sweep motion control in Fig. 6.

れらすべてを評価することは出来ない．また，複雑な経路を計画してもその動作の実現は困難である．

掃引開始当初では，地図情報の信頼性が低いことと計算コストを考え，周囲の地図情報を利用し，壁面反射時の回転角度を制御する方法を用いる．ここでは，経路の候補を衝突後の回転動作とその後の直進動作からなるものに限定する（図 2.12 参照）．これらの直進経路候補の中から有限時間 T においてエントロピーの和の値が最大となるものを選択する．選択する角度 ϕ を以下の式で計算する．

$$\phi = \arg\max_{j \in A} \sum_{\tau=1}^{T} H_p(m_i(\boldsymbol{x}_{j,t+\tau})) \tag{2.19}$$

ここで，A は候補となる角度の集合を表す．$m_i(j, \boldsymbol{x}_t)$ は，方向 j で時刻 t において予測されるロボットの位置姿勢 $\boldsymbol{x}_t$ が含まれる格子セルを表す．図 2.12 では地図が格子セルで表されており，濃い色の格子セルは障害物・壁である確率が高いことを表

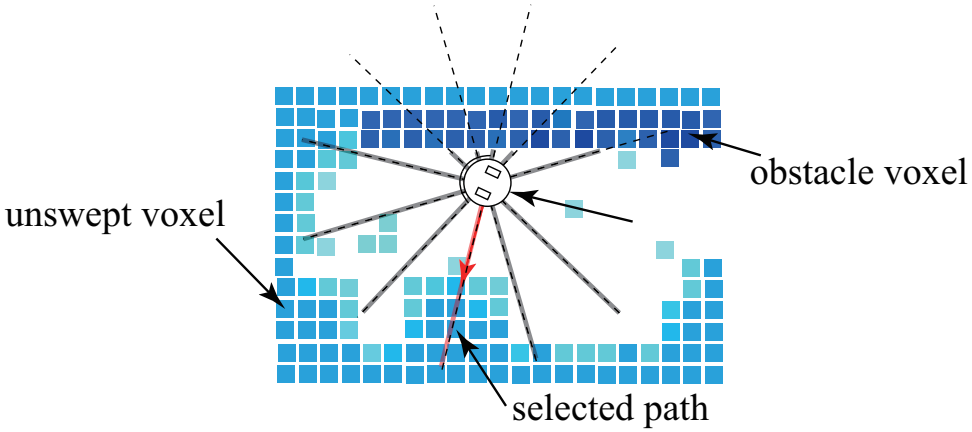


Fig. 2.12 This illustrates a control using local information. The blue boxes indicate obstacles, and the light blue boxes indicate unswept area. The robot selects a specific direction of the next motion which has biggest entropy. The robot rotates to the selected path direction, then move to the direction.

し，白色に近づくほど既掃引領域である確率が高いことを表す．中間の色の格子セルは未掃引領域であることを表している．

2.5.3　Global Reference-based 動作制御

図 2.10 の実線枠で示す大域情報を用いた動作制御を説明する．地図が漸進的に作成され，大域地図の信頼性が高まれば，自由空間と障害物，掃引と未掃引領域に関する大域情報を用いた効率的経路を生成し，これに基づく制御を行うことが望ましい．大域情報を用いた動作制御の手順は以下通りである（図 2.13 参照）．

1. セル m_i の確率的表現 $p(m_i)$ から障害物・壁領域，既掃引領域と未掃引領域を表す，式（2.20）の 3 値確定表現に変換する．

2. 大域的経路生成のため既掃引領域と未掃引領域を合わせて凸包を用いて凸領域化する．

3. 未掃引領域を未掃引コロニーとして凸領域化する．

4. 複数の未掃引コロニーを訪問する順番を決める探索問題を解くことで掃引順序を決定する．

5. 未掃引コロニー内の掃引経路と未掃引コロニー間の移動経路を生成し，組み合わせて全体経路とする．

6. 生成した経路に追従する．

1. を補足説明する．これまで地図は障害物・壁領域，未掃引領域，既掃引領域を認識するために確率的表現 $p(m_i)$ としていたが，大域的経路生成のため，これらを確定的表現に変換する．これは閾値を設定することでなされるが，掃引終盤ほどセル m_i のエントロピーが低いものも考慮する必要があるため，閾値を掃引途中で変化させることが望ましい．そのため，掃引率の見積もり $\hat{r}$ を計算し，$\hat{r}$ の値によって閾値を変化させることとする．掃引率の見積もり $\hat{r}$ は，広さが不明な環境に対してどの程度掃引が完了したかを生成した地図から推定する必要がある．これは一般に難しい問題のため，本稿では，本掃引戦略により図 2.1(iii) のように掃引が進んでいることを仮定し，時間に単調増加で $\hat{r} = 1$ に漸近収束する指数関数を掃引率の見積もり $\hat{r}$ として用いた．

既掃引領域と未掃引領域の閾値，未掃引領域と障害物・壁領域の閾値をそれぞれ $o_f(\hat{r})$，$o_w(\hat{r})$ と表し，掃引率の見積もり値 $\hat{r}$ の関数と表現すると，格子セル m_i を以下のように 3 値で表現する．

$$W(p(m_i), \hat{r}) \quad = \quad \begin{cases} 0 & (p(m_i) \leq o_f(\hat{r})) \\ 1 & (o_f(\hat{r}) < p(m_i) < o_w(\hat{r})) \\ 2 & (o_w(\hat{r}) \leq p(m_i)) \end{cases} \tag{2.20}$$

$W(p(m_i), \hat{r}) = 0$ のとき格子セル m_i は既掃引領域，$W(p(m_i), \hat{r}) = 1$ のときは未掃引領域，$W(p(m_i), \hat{r}) = 2$ のときは障害物・壁領域を意味する．

2. を補足説明する．凸分割においてはそれぞれの凸化された領域が大きくかつ数が少ないほうが，4 のグラフ探索で計算量が少なくなる．そのため，ロボットの推定軌跡上の複数離散点を中心として，放射状に一定距離で直線を延ばし，もし障害

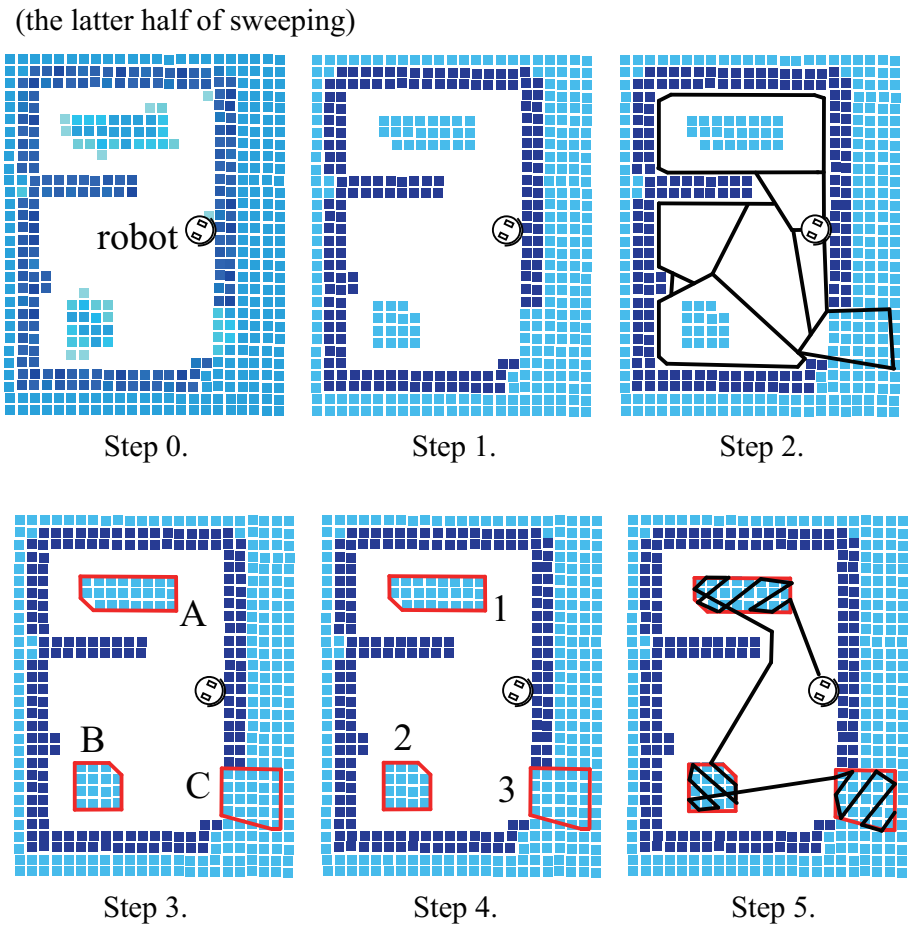


Fig. 2.13 These figures represent the sequence of the planning for a sweeping path using global information. Step 1) divides the map into three parts, obstacle area, swept area and unswept area. Step 2) decomposes into multiple convex areas. Step 3) recognizes the unswept colonies by a clustering method. Step 4) decides the order of visiting to the unswept colonies. Step 5) plans the path to move from an unswept colony to an another unswept colony, and also plans the path of inside the colony.

物・壁領域に到達すればそこで直線を延ばすことを停止する．延ばした直線の先で囲まれる領域を凸分割する手法を用いることとする．

3. を補足説明する．2. とは異なり，3. では未掃引領域を正確に把握することが必要である．そのため，クラスタリング処理を用いて未掃引領域であるすべての格子

セルを複数の塊へと分割する．クラスタリング手法としては，分割数が未知時に処理が簡単な k-Means 法 [91] を用いて，凸形状に分割を行っている．

2. にて凸化された領域内に存在する未掃引格子セル一つをデータ一つとみなし，クラスタリング処理をおこなう．ここで，塊の個数はわからないため，クラスタの数を固定にすることは危険である．3. では，まずデータ一つにクラスタ一つを割り当てておき，クラスタ間を結ぶ直線が未掃引領域に含まれる場合クラスタをまとめるという手法を取った．分割された未掃引領域を未掃引コロニーとする．

4.，5. に関しては次章で述べる手法を用いる．

なお，図 2.10 の IV の掃引終了判定は 2. で空間構造化された領域内の格子セルのエントロピーの和が閾値以下になることで判定する．

2.6　未掃引コロニー掃引経路計画手法

現実的な掃引ロボットを用いての掃引作業後半には，未掃引部分が散在して存在する状況が出来る．この状況で反射的動作のみでは掃引効率が著しく低下することになるため，効率的な掃引計画が是非必要である．ここでは，散在する未掃引部分に適当な凸化処理を行ったあとの状況を考え，図 2.13 の Step 3 のような複数の散在する凸領域すなわち，散在する未掃引コロニー（図 2.13 の A,B,C が未掃引コロニー）に対する効率的掃引問題を議論する．

2.6.1　複数の未掃引コロニー掃引問題

散在する未掃引コロニー掃引のための経路生成問題は 2 つの問題に分けることができると考えられる．1 つは未掃引コロニーを訪れる順序を決定する問題，もう 1 つは未掃引コロニー間移動経路とコロニー内掃引経路を生成する問題である．順序決定問題は複数の未掃引コロニーを訪問地と考えることで，1 つの巡回セールスマン問題とみなすことができる．このため，移動経路と掃引経路のコストを評価できれば，その巡回セールスマン問題を解くことで，未掃引コロニーを訪れる順序が決定

できる．ここで，コロニー内掃引経路は凸領域内で重複のない効率的掃引経路として知られている，水平パラレル動作 [89]（図 2.14 の未掃引コロニー U_i 内の経路）もしくは輪郭パラレル動作 [95]（図 2.14 の未掃引コロニー U_j 内の経路）を用いれば，経路が決定され，掃引経路コストを評価できる．

しかしながら，これらは未掃引コロニー入口点（図 2.14 では点 A や点 E など）によりコロニー内掃引経路が異なる．また，水平パラレル動作では入り口点が決まれば出口点（図 2.14 では点 B か点 F など）は自動的に決まるため，次のコロニーへの移動経路の長さに影響を与える．このことは，この散在する未掃引コロニー掃引問題では，コロニーへの入口点も変数として組み込んだ巡回セールスマン問題を解く必要があることを意味する．巡回セールスマン問題は一つの組み合わせ最適化問題なので，入口点を未掃引コロニー周囲を離散化することで対応するとしても，膨大な組み合わせ数となり，良い解を得ることは困難であることが予想される．実際には，入口点（輪郭パラレルではそれに加え出口点）を変化させても最終的な経路長に大きな影響を与えないとも考えられる．そこで，本研究ではこの問題を，未掃引コロニー順序決定問題と未掃引コロニー間移動経路＋コロニー内掃引経路生成問題に分離して，それぞれ別に解く準最適な解法を提案する．また，次節でこの解法が，同時に最適化する場合での結果と差がほとんどないことを示す．

2.6.2　未掃引コロニー巡回順序決定

未掃引コロニーを訪れる順序を決定する問題は，移動経路コストと掃引コストを考慮して，最小コストですべての未掃引コロニーを訪問する問題とみなすことができる．ここでは，コロニー間移動経路は各コロニー面心間最短経路（図 2.15 の破線）とし，コロニー内の線分を除いたものとする（例えば図 2.15 の d_{ij}）．また，コロニー内では重複ない経路で掃引できるものとして，コロニー U_i の面積 S_i，掃引幅 w，掃引速度 v により $T_i = \frac{S_i}{vw}$ で掃引に要する時間 T_i を見積もる．

コロニー U_i，コロニー U_j 間の巡回セールスマン問題でいう移動コスト C_{ij} を

$$C_{ij} = \frac{d_{ij}}{v} + \frac{S_i + S_j}{2vw} \tag{2.21}$$

で計算する．右辺前半の項 $\frac{d_{ij}}{v}$ は移動コストを表し，右辺後半の項 $\frac{S_i+S_j}{2vw}$ は，出発するコロニーの掃引コスト半分と到着するコロニーの掃引コスト半分を足し合わせたものとしている．C_{ij} は，コロニー U_i，コロニー U_j を掃引しながら移動することを含めた，面心間の移動時間を表している．

ここでそのまま，コスト最小となる経路を決定する探索問題（一種の巡回セールスマン問題）を解くと，コロニーの数が多い場合は計算量が大きくなる．本例題では掃引作業中にオンラインで大域的経路生成を行うため，計算時間の制約が大きい．また，遠くの計画経路まで追従制御する場合は，自己位置推定誤差が累積することにより，正確な掃引作業の遂行は困難であり，大域経路生成の意味を失う可能性が高い．そこで，ある掃引順路 α に対して，まず式（2.21）で経路所要時間を見積もる．そしてある時間 T_a までの経路に含まれる未掃引コロニーに限定し，その時間以降に到達する未掃引コロニーで獲得できる面積は 0 とすることで，遠くの未掃引コロニーは考慮しない．結局，以下の式により，ある時間 T_a までに掃引可能な面積が最も大きくなる順路を評価する経路探索問題を解くことで，コロニー間掃引経路（すなわち巡回順序）とする．

$$S_\alpha(T_a) = \sum_{i=0}^{n} \omega_i(t_i^{in}, t_i^{out}, T_a) S_i \tag{2.22}$$

ここで，S_α はある順路 α で時刻 T_a までに獲得できる掃引面積を表し，S_i は順路 α で i 番目に掃引する未掃引コロニーの面積を表している．$\omega_i(t_i^{in}, t_i^{out}, T_a)$ は，t_a が掃引開始時刻 t_i^{in} と掃引終了時刻 t_i^{out} で表され，$T_a < t_i^{in}$ のときには $\omega_i = 0$，$t_i^{out} < T_a$ のときには $\omega_i = 1$，$t_i^{in} \leq T_a \leq t_i^{out}$ のときには $\omega_i = (T_a - t_i^{in})/(t_i^{out} - t_i^{in})$ となる関数とする．掃引開始時刻 t_i^{in} と掃引終了時刻 t_i^{out} は未掃引コロニー内で掃引にかかる時間と，面心間の移動にかかる時間より計算することができる．

2.6.3　未掃引コロニー間とコロニー内の掃引経路生成

提案の経路生成法での掃引開始点と掃引終了点の決定方法を図 2.15 に示す．$P_{in,i}$，$P_{out,i}$，$P_{c,i}$ はそれぞれ未掃引コロニー U_i の掃引開始点，掃引終了点，面心点を表す．破線は未掃引コロニーの掃引順路決定時に計画された，面心間を結ぶ経路を表す．掃

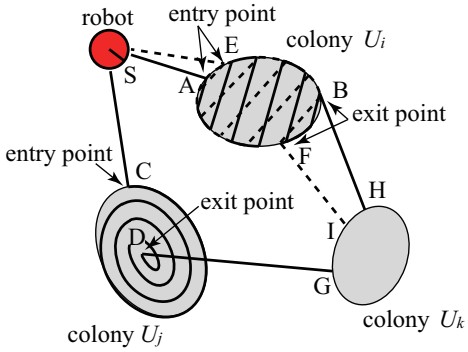


Fig. 2.14 The directional-parallel (the path of the unswept colony U_i) and the contour-parallel (the path of the unswept colony U_j) are known as typical optimal sweeping path for convex area. The sweep exit point of an unswept colony is uniquely determined according to the sweep entry point by an optimal planning method in the unswept colony. The efficiency of the optimal sweeping path depends on the selection of the entry point. Therefore, the robot must find the "efficient entry point" for the unswept colonies.

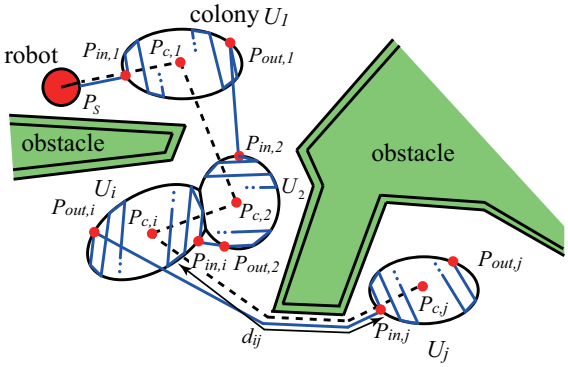


Fig. 2.15 This figure illustrates how to select the entry point. For the first unswept colony, the entry point is selected as the cross point $P_{in,1}$ of the colony boundary and the line from the robot to the center of the colony. After finding the exit point of the colony using the optimal motion planning inside of the colony, the path to the next unswept colony is selected as a line of the shortest distance to the next unswept colony. Repeat this procedure until final unswept colony. The method does not execute the exact optimization of the selection for the entry point, because the optimization takes so much time and resultant path by the proposed method is almost same with the optimized one.

引終了点は掃引開始点が決まれば，水平パラレル動作により一意に決定できるとすると，領域内の掃引経路と領域間の経路を組み合わせた経路が最終経路となる．

掃引開始点・掃引終了点と未掃引コロニー内の経路は順に決定される．まず，ロボットの初期位置 P_s と U_1 の面心 $P_{c,1}$ との最短経路を求め，その最短経路と U_1 の外形との交点を掃引開始点 $P_{in,1}$ とする．掃引終了点 $P_{out,1}$ は $P_{in,1}$ により一意に決定される．それ以降の掃引開始点 $P_{in,i}$ は，U_{i-1} の掃引終了点 $P_{out,i-1}$ と面心点 $P_{c,i}$ の最短経路を求め，その最短経路と U_i の外形の交点とする．掃引終了点 $P_{out,i-1}$ は先ほ

どと同様に，$P_{in,i}$ より一意に決定される．

凸領域内の掃引経路は，経路長やロボットの回転量などを考慮して評価するが，本稿では議論を行わない．なぜなら，凸領域内掃引アルゴリズムの差が明確となるような面積の大きい未掃引領域では，未発見の障害物が存在している確率が高く，そのような際には，新たに掃引経路計画をおこなう必要があるためである．

生成した掃引コロニー内の掃引経路と未掃引コロニー間の移動経路を合わせたものが，大域情報を用いた動作制御により計画した経路となる．

2.7　シミュレーションと実験

提案する漸進的生成地図に基づく掃引動作制御手法が，従来の主に反射的行動を主体とする手法に比べ有効であるか検証するため，シミュレーションおよび実ロボットを用いた実験を行った．ロボットが一度以上覆った領域を掃引されたとみなし，既掃引率を理論的掃引可能な全自由領域に対する既掃引領域の割合としている．

2.7.1　シミュレーション

ロボットのオドメトリ誤差として並進速度 v および回転速度 ω には実機を想定した誤差を加える．並進速度 v と回転速度 ω のオドメトリ誤差は平均 0 のガウス分布で共分散 $\sqrt{\alpha_1 v^2 + \alpha_2 \omega^2}$ と $\sqrt{\alpha_3 v^2 + \alpha_4 \omega^2}$ [16] に従うものとした．係数は実際の実験でのオドメトリ誤差をもとに Table 1 に示す値を設定した．シミュレーションは図 2.16 に示す 4 つの部屋形状で 30 回ずつ行った．以降で述べる行動ベース手法とは，掃引開始時から壁に衝突するまで螺旋運動を行い，その後の動作時間もしくは衝突回数の条件によって壁面追従動作と壁反射動作（回転角はランダムで選択）を繰り返す動作制御手法のことである．

シミュレーションは以下の 2 種類行った．1 つ目として，本研究で提案している漸進的生成地図情報を用いる掃引動作制御手法が，反射的行動のみに基づく手法に比べて，掃引効率が高いかどうかのシミュレーションを行う．このため，特徴が出や

すい掃引後半の掃引率の変化に注目し，図 2.17 のような掃引途中の状況を想定し，この状況で掃引を開始した時点からの掃引率の変化を調べる．提案手法においては漸進的地図生成を行っているため，この図 2.17 に示す生成途中地図を与え，以降の掃引動作生成を行った．

シミュレーションによる 30 回の試行から計算した時刻 0 分から 10 分のそれぞれの時間の平均既掃引率を図 2.18 に示している．それぞれ，実線が提案する準最適な解法を用いる手法の結果，破線が計算量の多い最適な解法を用いる手法の結果，1 点鎖線は局所情報のみを用いた動作制御手法の結果，2 点鎖線は行動ベース手法の結果である．図 2.18(a) の提案する動作制御の結果は図 2.1 で示した (iv) の期待する掃引率のグラフと類似していると言える．その他の部屋形状では (a) の部屋形状よりも緩やかであるが，これは未掃引コロニー間の距離が長くなったことによる掃引率の低下と考えられる．局所情報のみ用いた動作制御を行うロボットは，区切られたような部屋形状 (d) で他の部屋形状に比べ掃引率の低下が見られた．

また，前章で述べた計算量の多い最適解法を行った動作制御と，提案する準最適解法を用いた動作制御を比較すると，全ての部屋形状において掃引率はほとんど変わらないことがわかる．このことより準最適な解法を用いることは妥当である．準最適解法と最適解法では計算時間に 30 倍程度違いがあった．なお，準最適解法では CPU：Intel Core2 Duo 2.33GHz　メモリ：3.25GB の PC を用いた場合，平均で 30 秒程度要した．プログラムの十分な最適化と並列化を行うことで，数秒程度で計算が可能だと考えられる．数秒程度であればロボットが動作しながら，計算を行うことでリアルタイムな動作制御が可能である．一部準最適解法の結果が最適解法の結果を上回っている（図 2.18（b）,（c））．これは地図の曖昧さやロボットの制御誤差によるものである．このような確率要素が大きくなるにつれ,（曖昧な地図に対して）最適な解法は実環境での最適性を失っていく．

2 つ目のシミュレーションは，部屋全体を最初から掃引する状況を想定して，提案手法と行動ベース手法の掃引率の変化を調べる．図 2.19 にこの場合のシミュレーション結果を示す．実線が提案手法，破線が行動ベースを行うロボットの結果である．図 2.19 の縦線は標準偏差を表している．地図や自己位置など確率的な要因によ

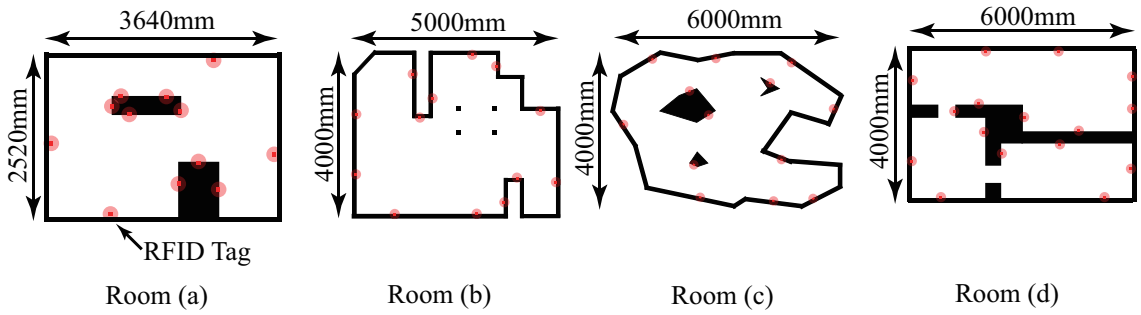


Fig. 2.16 Four rooms with different shapes tested in the simulation. The red circles represent the RFID tags.

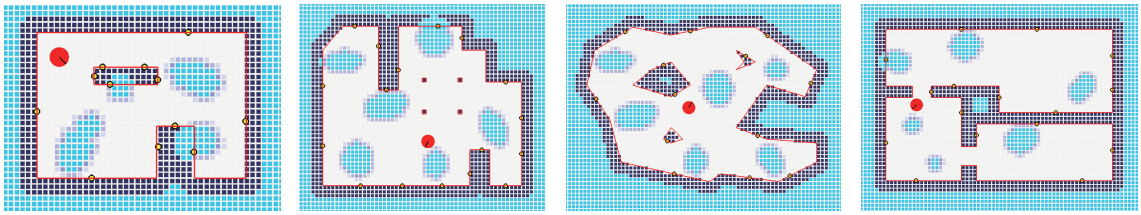


Fig. 2.17 Red circle in the figures represents the sweeping robot. The dark blue area shows the estimated wall, and the light blue area shows the unswept area. These figures show sweeping situation in the middle of sweeping task.

り図 2.18 のシミュレーション結果よりも効率が低下しているが，提案手法が有効であることが確認できる．標準偏差の時間変化に着目すると，本提案アルゴリズムでは，掃引が進むにつれ，標準偏差が小さくなっていることがわかる．掃引開始時は，環境情報が乏しいことに加え，スタート地点がランダムであることから掃引率の偏差が大きいとみられ，環境情報が得られるにつれ，効率的な掃引ができていると考えられる．

2.7.2　実験

実験は，市販の掃除ロボット iRobot 社 Roomba を改造し，交換した制御基板（主として Renesas Technology 社 H8/3052F で構成）によりセンサとモータを制御した

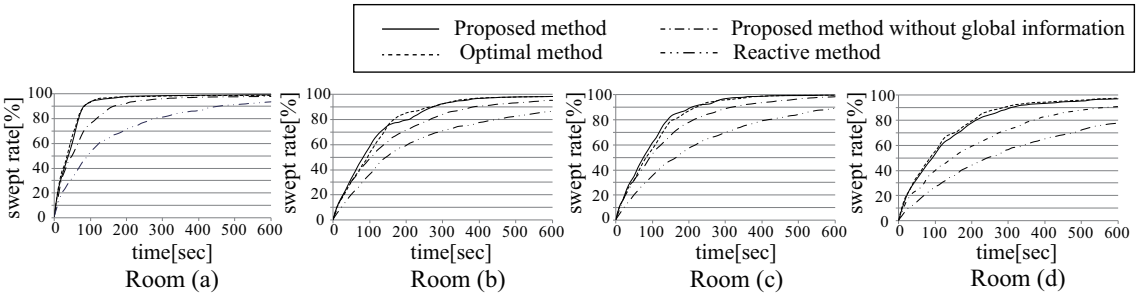


Fig. 2.18 Swept rate for various sweeping control method (simulation1). The solid line, the dotted line, the dashed line and the two-dot line respectively represents the swept rate results of the proposed method, the optimal method, the proposed method without global information and the reactive method.

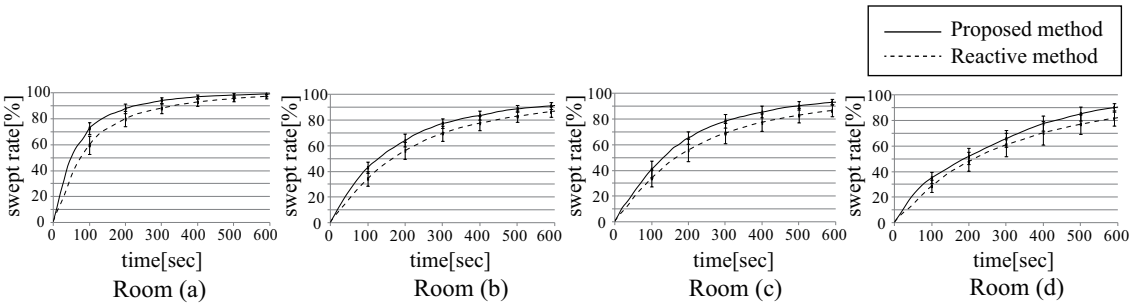


Fig. 2.19 Swept rate and standard deviation by proposed and reactive method (simulation2). The dotted line shows the mean swept rate of the robot using the reactive method, and the solid line indicates the motion using the proposed method.

もの（図 2.20（a））を用いて行った．また，本例題のモデルと同様に RFID リーダアンテナを取り付けている．RFID タグの側をロボットが通過すると，RFID リーダアンテナを通してタグの位置姿勢を取得することができるロボットにはシリアル-Bluetooth モジュールを搭載しており，掃引中は PC の Bluetooth ドングルと通信している．ロボットはセンサ情報を PC に送信し，PC（CPU：Intel Core2 Duo 2.33GHz

メモリ：3.25GB）で軌道推定と地図生成，動作計画を行った後，ロボットに動作指令を送信する．表 2.1 にロボットのパラメータを示す．

図 2.21 に示す 2 つの部屋形状で実験を行った．それぞれサイズ 2520mm × 3640mm

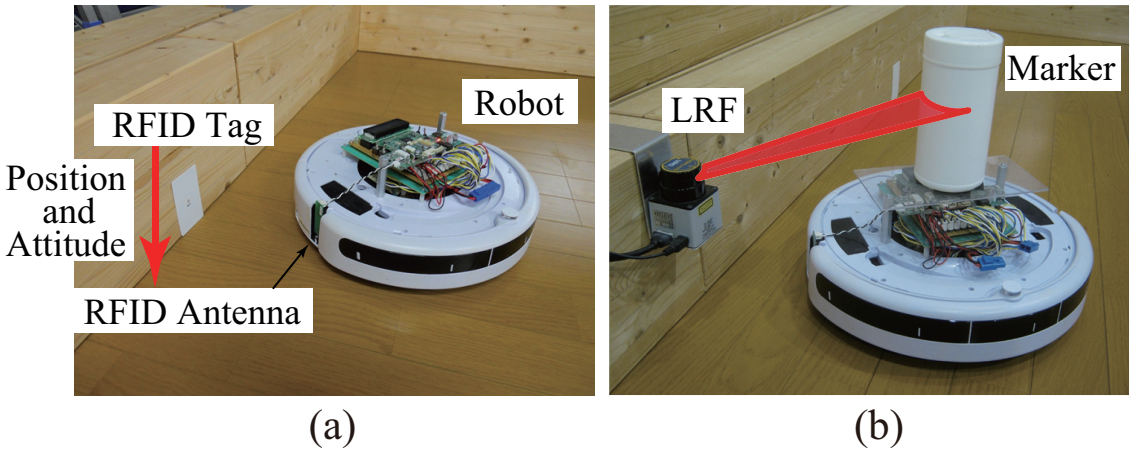


Fig. 2.20 Sweeping robot for the experiment.

Table 2.1 Parameters of the sweeping robot

Diameter	34 [cm]
Translational velocity	30 [cm/s]
Rotational velocity	90 [deg/s]
Odometry error α_1〜α_4	0.03

の空間に障害物を置き，複数のタグを壁に取り付けている．掃引率を計測するため，図 2.20（b）のように LRF（HOKUYO URG-04LX）を壁に取り付け，ロボットに取り付けたマーカを検出し，ロボットの中心を計算することでロボットの真の位置とした．ただし，LRF は計測のみで使用し，ロボットの制御には使用していない．姿勢は，ロボットの初期の移動から求めた初期姿勢と，ロボットの真の位置から拡張カルマンフィルタを用いて推定している．

実験による 10 分間の掃引率の変化を図 2.22 に示している．実線が提案手法，破線が行動ベースを行うロボットの結果である．ランダムにロボットの初期姿勢（位置と角度）を変化させた 5 回の試行から計算した時刻 0 分から 10 分のそれぞれの時刻の平均既掃引率とその標準偏差を示す．性能は図 2.19 のシミュレーションと比べわずかに劣るが，いずれの部屋でも掃引率の向上が確認された．また，標準偏差を用いて 2 つの動作制御を比較しても，提案する動作制御の方が有効であることがわ

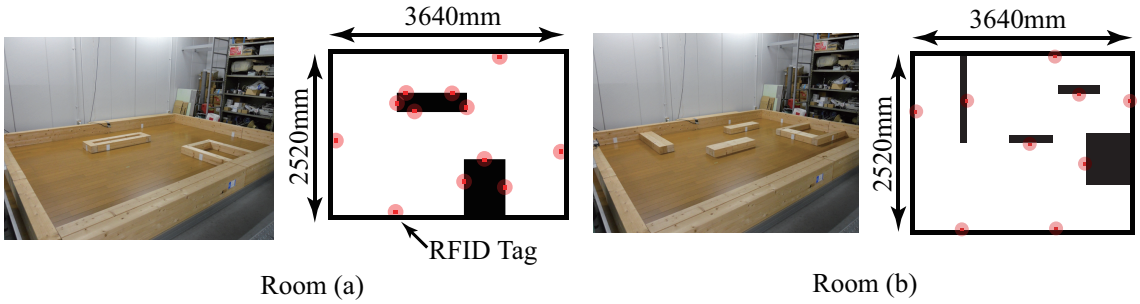


Fig. 2.21 Two different room shapes for the experiment. The red circles represent the RFID tags.

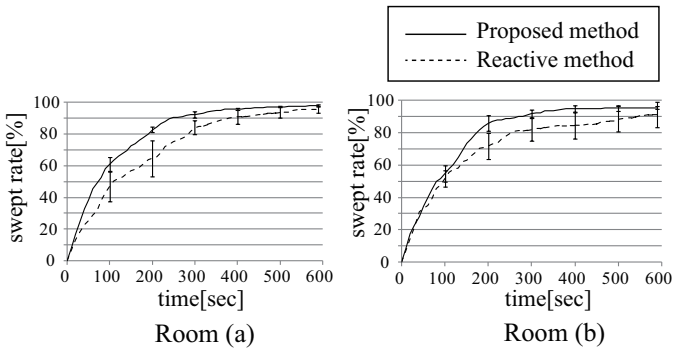


Fig. 2.22 Swept rate and standard deviation by proposed and reactive method (experiments). The dotted line indicates the mean swept rate of the robot using the reactive method, and the solid line indicates the robot using the proposed method.

かる．

2.8　GBM-based アルゴリズム概念図の導出

本節では，時刻と掃引率の関係，3 つの動作制御とその遷移する確率から，ある掃引率における 3 つの動作制御が選択される確率を求める．これは，完全原因系に対するベイズの公式（式（2.27））を用いて求めることができる．ある時刻における LR，GR，LS が選択される確率から，掃引戦略図を得ることができる．

2.8.1　定義と仮定

ここでは，本節で用いる公式，記号の定義と仮定について説明する．

完全原因系に対するベイズの公式

Ω を標本空間，A をある事象とする．また，$n \geq 1$ としたとき，B_n を相互に排反な事象とする．$i \neq j$ のとき，常に $B_i \cap B_j = \phi$ であり，$\bigcup_{n=1}^{\infty} B_n = \Omega$ を意味する．この時，ベイズの公式は以下の式で表される [96].

$$P(A) = \sum_{n=1}^{\infty} P(A|B_n)P(B_n) \tag{2.23}$$

これを完全原因系に対するベイズの公式と呼ぶ.

今 3 つの事象 A, B, C を考える．ベイズの事象列公式 [96] から，

$$\begin{aligned} P(A \cap B \cap C) &= P(A|B \cap C)P(B|C)P(C) \\ &= P(B|A \cap C)P(A|C)P(C) \\ &= P(C|A \cap B)P(A|B)P(B) \end{aligned} \tag{2.24}$$

B と C が独立であるとすると，上式の 1 列目と 2 列目の右辺から

$$P(B|A \cap C)P(A|C) = P(A|B \cap C)P(B) \tag{2.25}$$

3 つの確率変数を X, Y, Z，Y と Z は独立であるとし，$x \in X, z \in Z$ とする．$y \in Y$ は，$\{0, \mathbb{R}^+\}$ の値を取るとする．式（2.25）は，確率変数にも同様に成り立つので，

$$\begin{aligned} &P(Y = y|X = x, Z = z)P(X = x|Z = z) \\ &\qquad = P(X = x|Y = y, Z = z)P(Y = y) \end{aligned} \tag{2.26}$$

この時，完全原因系に対するベイズの公式は，

$$P(X = x|Z = z) = \int_0^{\infty} P(X = x|Y = y, Z = z)P(Y = y)dy \tag{2.27}$$

となる.

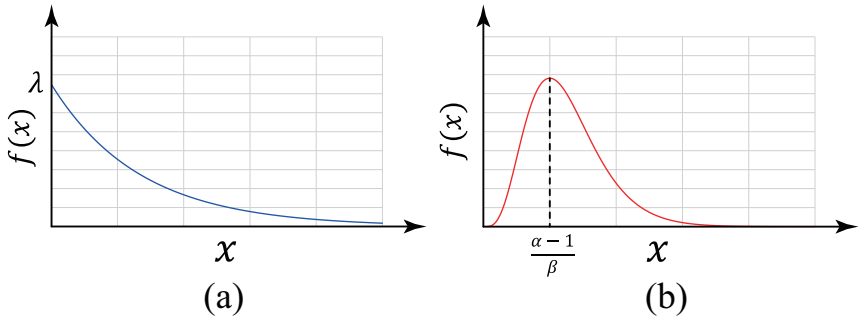


Fig. 2.23 Probability density function

確率密度関数

確率密度関数 f は，次式を満たさなければならない [96].

$$\int_{-\infty}^{+\infty} f(y)dy = 1 \tag{2.28}$$

パラメータ $\lambda > 0$ の指数分布の確率密度は，

$$f(x) = \begin{cases} \lambda e^{-\lambda x} & (x \geq 0) \\ 0 & (otherwise) \end{cases} \tag{2.29}$$

$\lambda > 0$ のときの確率密度の形を図 2.23（a）に示している．

α, β を正の実数とし，ガンマ確率密度 [96] を次のように定義する．

$$f(x) = \begin{cases} \dfrac{\beta^{\alpha}}{\Gamma(\alpha)} x^{\alpha-1} e^{-\beta x} & (x > 0) \\ 0 & (otherwise) \end{cases} \tag{2.30}$$

ただし，$\Gamma(\alpha)$ はガンマ関数で

$$\Gamma(\alpha) = \int_0^{\infty} u^{\alpha-1} e^{-u} du \tag{2.31}$$

パラメータ $\alpha > 1$ のとき，確率密度の形は図 2.23（b）のようになる．

各動作制御が選択される確率モデル

まず初めに，3 つの確率変数 $\mathcal{S}, \mathcal{R}, \mathcal{E}$ を定義する．状態を表す確率変数 $\mathcal{S}$ は，ロボットが選択している 3 つの動作制御 $\{LR, GR, LS\}$ の値のどれかをとる．自己位置推定誤差を表す確率変数 $\mathcal{E}$ は，$[0, \infty]$ から値をとり，掃引率を表す確率変数 $\mathcal{R}$ は，$[0, 1]$ の中の値をとる．

次に誤差 ε に関する仮定をたてる．ロボットは，停止することなく動作し続けるものとし，動作によってロボットの自己位置推定誤差は，増加することから，時刻 t によって線形に増加するものとする．つまり，

$$\varepsilon(t) = k_\varepsilon(t - t_c) \tag{2.32}$$

ここでの，k_ε は誤差の増加率，t_c は最後にランドマークを検出し，自己位置修正を行った時刻とする．t_c は掃引率 r との関係性はないため，誤差 ε と掃引率 r は独立である．

誤差が $\mathcal{E} = \varepsilon$，掃引率が $\mathcal{R} = r$ であるときに，それぞれの状態が選択される確率を仮定する．GBM-based アルゴリズムは，ロボットの自己位置推定誤差が大きいときには，優先的に LS を選択する．LS が選択される確率は，誤差 ε が大きくなるにつれ，大きくなる．そこで，次式でモデル化する．

$$P(\mathcal{S} = LS | \mathcal{E} = \varepsilon, \mathcal{R} = r) = 1 - \lambda_e e^{-\lambda_e \varepsilon} \tag{2.33}$$

パラメータ λ_e は，$0 \leq \lambda_e < 1$ を満たす．誤差がない状態，つまり $\lambda_e = 0$ を考えた時，$P(\mathcal{S} = LS | \mathcal{E} = \varepsilon, \mathcal{R} = r) = 0$ となり，LS は選択されないことを意味する．

ロボットの自己位置推定誤差が小さい場合，GBM-based アルゴリズムの次の条件分岐,「ロボットが周辺の地図情報から未掃引領域を発見できるか」の判定により LR か GR かが選択される．

掃引率の増加量が大きな時には，周辺に未掃引領域が存在する可能性は大きくなること，掃引率が大きくなるにつれて，周辺の未掃引領域は発見されにくくなること，という 2 点から，LR が選択される確率を次式のようにモデル化する．

$$P(\mathcal{S} = LR | \mathcal{E} = \varepsilon, \mathcal{R} = r) = \lambda_e e^{-\lambda_e \varepsilon} \quad \varphi(r, \dot{r}) \tag{2.34}$$

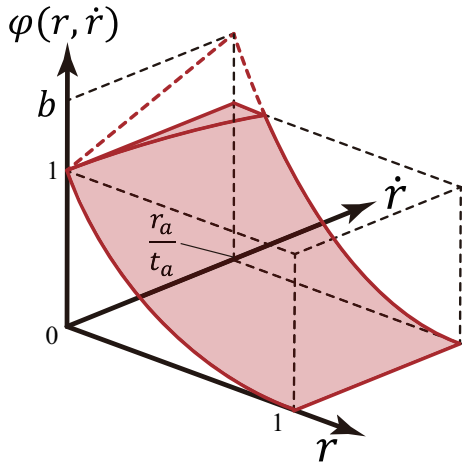


Fig. 2.24 This graph shows the conditional probability distribution $\varphi(r,\dot{r})$ of $\mathcal{S}=LR$ given as $\mathcal{E}=\varepsilon,\mathcal{R}=r$.

ここで，$\varphi(r,\dot{r})$ は図 2.24 に示すような形を持つ関数で，例として

$$\begin{aligned}\varphi(r,\dot{r}) &= \max\{1,\ \hat{\varphi}(r,\dot{r})\} \\ \hat{\varphi}(r,\dot{r}) &= \left(\frac{(b-1)t_a}{r_a}\dot{r}+1\right)\left(1-\sqrt{-r(r-2)}\right)\end{aligned} \tag{2.35}$$

のような式が挙げられる．$b>1$ とする．

一方，GR が選択される確率は，

$$P(\mathcal{S}=GR|\mathcal{E}=\varepsilon,\mathcal{R}=r)=\lambda_e e^{-\lambda_e\varepsilon}\ \ (1-\varphi(r,\dot{r})) \tag{2.36}$$

と表される．

上式はすべての $\mathcal{E}=\varepsilon$，$\mathcal{R}=r$ において，

$$\begin{aligned}P(\mathcal{S}=LS|\mathcal{E}=\varepsilon,\mathcal{R}=r)+P(\mathcal{S}=LR|\mathcal{E}=\varepsilon,\mathcal{R}=r)\\ +P(\mathcal{S}=GR|\mathcal{E}=\varepsilon,\mathcal{R}=r)=1\end{aligned} \tag{2.37}$$

を満たしている．

誤差 ε の確率密度

完全原因系に対するベイズの公式（式（2.27））を用いて，ロボットの位置誤差 $\mathcal{E}$ によらない確率を求める．3 つの確率変数 $\mathcal{S},\mathcal{R},\mathcal{E}$ を $\mathcal{S}=s,\mathcal{R}=r$ として，式（2.27）

に当てはめると，

$$P(\mathcal{S} = LS|\mathcal{R} = r) = \int_0^\infty P(\mathcal{S} = s|\mathcal{E} = \varepsilon, \mathcal{R} = r)P(\mathcal{E} = \varepsilon)d\varepsilon \tag{2.38}$$

$P(\mathcal{S} = s|\mathcal{E} = \varepsilon, \mathcal{R} = r)$ に関するモデルは，先に仮定した．$P(\mathcal{S} = LS|\mathcal{R} = r)$ を求めるためには，誤差 $\mathcal{E}$ が ε のときの確率 $P(\mathcal{E} = \varepsilon)$，つまり確率密度が必要となる．

ロボットの自己位置の不確かさは，誤差の大きなセンサ情報の取得（時間更新）により大きくなる．さらに，ロボットが LS を選択したとしても，改善までには時間を要し，この時間は一定ではない．誤差 $\mathcal{E}$ がゼロとなることはない点と，誤差 $\mathcal{E}$ は LS を選択するための条件（図 2.5（a）Check2）の閾値により，ある値付近で大きな確率になる点を考慮すると，誤差 $\mathcal{E}$ は，$\alpha > 1, \beta > 0$ としたガンマ確率変数としてモデル化することができる．

$$\mathcal{E} \ \sim \ \gamma(\alpha, \beta) \tag{2.39}$$

2.8.2　各掃引動作制御の遷移モデル

ロボットの位置誤差 $\mathcal{E}$ によらない確率 $P(\mathcal{S} = LR|\mathcal{R} = r)$，$P(\mathcal{S} = GR|\mathcal{R} = r)$，$P(\mathcal{S} = LS|\mathcal{R} = r)$ を求めることである．ある時刻における LR，GR，LS が選択される確率から，GBM-based アルゴリズムの掃引戦略図（図 2.5（b））を得る．

始めに，掃引率が $\mathcal{R} = r$ であるときの状態 $\mathcal{S}$ が LS である確率を求める．

$$P(\mathcal{S} = LS|\mathcal{R} = r) = \int_0^\infty P(\mathcal{S} = LS|\mathcal{E} = \varepsilon, \mathcal{R} = r)P(\mathcal{E} = \varepsilon)\ d\varepsilon \tag{2.40}$$

誤差 $\mathcal{E}$ は，ガンマ確率変数であると仮定した．よって，

$$\begin{aligned} P(\mathcal{S} = LS|\mathcal{R} = r) &= \int_0^\infty P(\mathcal{S} = LS|\mathcal{E} = \varepsilon, \mathcal{R} = r)P(\mathcal{E} = \varepsilon)\ d\varepsilon \\ &= \int_0^\infty (1 - \lambda_e e^{-\lambda_e \varepsilon})\ \frac{\beta^\alpha}{\Gamma(\alpha)} \varepsilon^{\alpha-1} e^{-\beta\varepsilon} d\varepsilon \\ &= \frac{\beta^\alpha}{\Gamma(\alpha)} \int_0^\infty \varepsilon^{\alpha-1} e^{-\beta\varepsilon} - \lambda_e \varepsilon^{\alpha-1} e^{-(\lambda_e+\beta)\varepsilon} d\varepsilon \end{aligned} \tag{2.41}$$

ここで，$p > 1$，$q > 0$ のとき

$$\int_0^\infty \varepsilon^{p-1} e^{-q\varepsilon} d\varepsilon = \frac{\Gamma(p)}{q^p} \tag{2.42}$$

なので，

$$\begin{aligned} P(\mathcal{S} = LS|\mathcal{R} = r) &= \frac{\beta^\alpha}{\Gamma(\alpha)}\Big(\frac{\Gamma(\alpha)}{\beta^\alpha} - \frac{\lambda_e \Gamma(\alpha)}{(\lambda_e+\beta)^\alpha}\Big) \\ &= 1 - \frac{\lambda_e \beta^\alpha}{(\lambda_e+\beta)^\alpha} \end{aligned} \tag{2.43}$$

と求められる．

同様に，掃引率が $\mathcal{R} = r$ であるときの状態 $\mathcal{S}$ が LR である確率を求める．

$$\begin{aligned} P(\mathcal{S} = LR|\mathcal{R} = r) &= \int_0^\infty P(\mathcal{S} = LR|\mathcal{E} = \varepsilon, \mathcal{R} = r) P(\mathcal{E} = \varepsilon)\, d\varepsilon \\ &= \int_0^\infty \lambda_e e^{-\lambda_e \varepsilon}\ \varphi(r, \dot{r})\ \frac{\beta^\alpha}{\Gamma(\alpha)} \varepsilon^{\alpha-1} e^{-\beta\varepsilon} d\varepsilon \\ &= \frac{\lambda_e \beta^\alpha}{\Gamma(\alpha)} \varphi(r, \dot{r}) \int_0^\infty \varepsilon^{\alpha-1} e^{-(\lambda_e+\beta)\varepsilon} d\varepsilon \\ &= \frac{\lambda_e \beta^\alpha}{\Gamma(\alpha)} \frac{\Gamma(\alpha)}{(\lambda_e+\beta)^\alpha} \varphi(r, \dot{r}) \\ &= \frac{\lambda_e \beta^\alpha}{(\lambda_e+\beta)^\alpha} \varphi(r, \dot{r}) \end{aligned} \tag{2.44}$$

掃引率が $\mathcal{R} = r$ であるときの状態 s が GR である確率は，

$$\begin{aligned} P(\mathcal{S} = GR|\mathcal{R} = r) &= \int_0^\infty P(\mathcal{S} = GR|\mathcal{E} = \varepsilon, \mathcal{R} = r) P(\mathcal{E} = \varepsilon)\, d\varepsilon \\ &= \int_0^\infty \lambda_e e^{-\lambda_e \varepsilon}\ (1 - \varphi(r, \dot{r}))\ \frac{\beta^\alpha}{\Gamma(\alpha)} \varepsilon^{\alpha-1} e^{-\beta\varepsilon} d\varepsilon \\ &= \frac{\lambda_e \beta^\alpha}{\Gamma(\alpha)} (1 - \varphi(r, \dot{r})) \int_0^\infty \varepsilon^{\alpha-1} e^{-(\lambda_e+\beta)\varepsilon} d\varepsilon \\ &= \frac{\lambda_e \beta^\alpha}{(\lambda_e+\beta)^\alpha} (1 - \varphi(r, \dot{r})) \end{aligned} \tag{2.45}$$

となる．

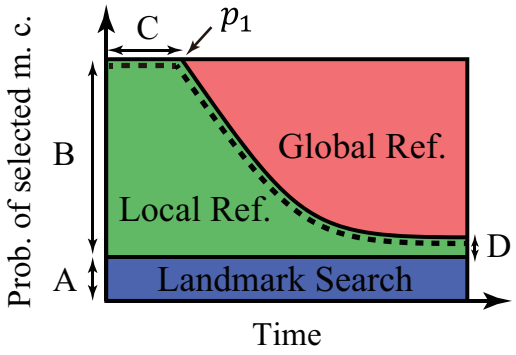


Fig. 2.25 The boundary and the parameters of the figure which represents the proposed algorithm's strategy.

2.8.3　掃引戦略図に関する考察

$\mathcal{R} = r$，$\mathcal{E} = \varepsilon$ のときに，LR が選択される確率 $P(\mathcal{S} = LR|\mathcal{E} = \varepsilon, \mathcal{R} = r)$ に，式（2.36）を例として掃引戦略図の各動作制御の境界について考察する．図 2.25 の A〜D，点線を求める．

$P(\mathcal{S} = LS|\mathcal{R} = r)$ は，時間によらず一定なので，図 2.25 の A は，

$$A = 1 - \frac{\lambda_e \beta^\alpha}{(\lambda_e + \beta)^\alpha} \tag{2.46}$$

であることがわかる．一方，$B = 1 - A$ であり，

$$B = \frac{\lambda_e \beta^\alpha}{(\lambda_e + \beta)^\alpha} \tag{2.47}$$

で表現される．

図 2.25 の破線は，式（2.44）

$$P(\mathcal{S} = LR|\mathcal{R} = r) = \frac{\lambda_e \beta^\alpha}{(\lambda_e + \beta)^\alpha} \varphi(r, \dot{r})$$

を時間の関数で表現したものに他ならないので，$\varphi(t)$ を求める．$\varphi(t)$ は，式（2.36）に式（2.1）を代入すると，

$$\varphi(t) = \max\{1, \hat{\varphi}(t)\} \tag{2.48}$$

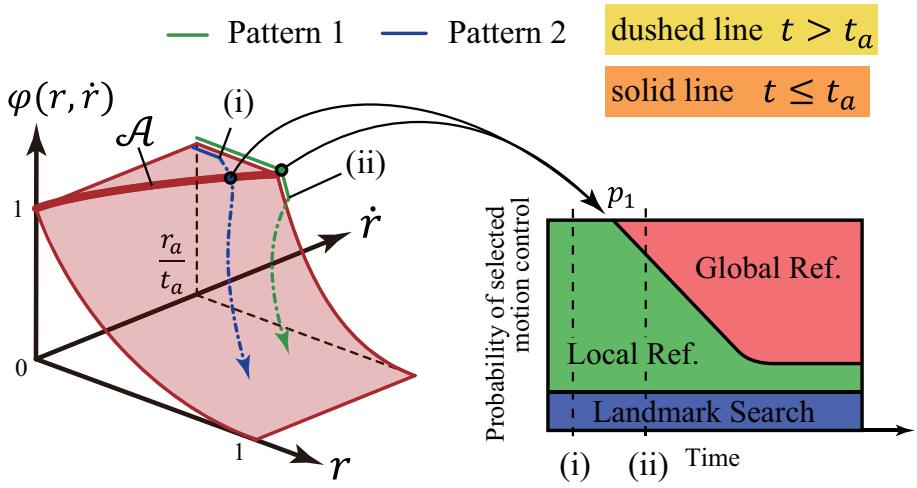


Fig. 2.26 The relation between the function $\varphi(r, \dot{r})$ and the figure which represents the proposed algorithm's strategy.

$\hat{\varphi}(t)$ は場合分けが必要で，$t \leq t_a$ の時

$$\hat{\varphi}(t) = b\left(1 - \sqrt{\frac{r_a}{t_a} t \ (2 - \frac{r_a}{t_a} t)}\right) \tag{2.49}$$

$t > t_a$ の時

$$\hat{\varphi}(t) = \left\{\frac{(r_m - r_a)t_a}{r_a}(b-1)\kappa e_{xp} + 1\right\} \cdot \left\{1 - \sqrt{(-(r_m - r_a)e_{xp} + r_m - 2)((r_m - r_a)e_{xp} - r_a)}\right\} \tag{2.50}$$

ただし，$e_{xp} = e^{-\kappa(t-t_a)}$ を意味する．

C の大きさを求める前に図 2.25 の p_1 について考える．図 2.26 に，関数 $\varphi(r, \dot{r})$ と掃引戦略図の関係を示している．図 2.26 の関数 $\varphi(r, \dot{r})$ 上に描かれている 2 本の線（緑線と青線）は，掃引ロボットが時間が進むにつれて通過する軌跡を表している．p_1 は GR の選択される確率が現れ始める点で，次の領域 $\mathcal{A}$ に存在することになる．

$$\mathcal{A} := \{(r, \dot{r}, \varphi) | \hat{\varphi}(r, \dot{r}) = 1, r \in [0, 1), \dot{r} \in (0, r_a/t_a]\} \tag{2.51}$$

ロボットが掃引を開始してから，領域 $\mathcal{A}$ に到達するまでの時間が C の大きさを表す．また，図 2.26 の（i）と（ii）は実線と破線の境目を意味し，時間と掃引率のモ

デル（図 2.2，式（2.1））の $t = t_a$ となる点を表している．式（2.1）の通り，実線は時間 t が $t \leq t_a$ の範囲で，掃引速度 $\dot{r}$ は r_a/t_a にあたる．青線の軌跡では，$\mathcal{A}$ に到達する以前に，$t = t_a$ となり，$t > t_a$ で領域 $\mathcal{A}$ に到達する．一方，緑線の軌跡では，$t \leq t_a$ の間に，$\mathcal{A}$ に到達する．このように C には 2 つのパターンが存在する．軌跡が（i）となる条件は，$t = t_a$ のときに $\hat{\varphi} > 1$ となることなので，b と r_a の関係式

$$b > \frac{1}{1 - \sqrt{r_a(2 - r_a)}} \tag{2.52}$$

で表せ，また軌跡が（ii）となる条件は，

$$b < \frac{1}{1 - \sqrt{r_a(2 - r_a)}} \tag{2.53}$$

とできる．

$t > t_a$ のとき $\hat{\varphi}(t) = 1$ となるのは，

$$t = \frac{t_a}{r_a}\left\{1 \pm \sqrt{\frac{1}{b}\left(2 - \frac{1}{b}\right)}\right\} \tag{2.54}$$

であり，これが C の長さとなる．

$$C = \frac{t_a}{r_a}\left\{1 \pm \sqrt{\frac{1}{b}\left(2 - \frac{1}{b}\right)}\right\} \quad (if\ Eq.(2.53)) \tag{2.55}$$

$t \leq t_a$ のときも同様に求められる．

最終的に LR が選択される確率 D は，$\varphi(t)$ の時間の極限をとることで得られる．

$$\begin{aligned} D &= \lim_{t\to\infty} \varphi(t) \\ &= 1 \cdot \left\{1 - \sqrt{(r_m - 2)(-r_m)}\right\} \\ &= 1 - \sqrt{r_m(2 - r_m)} \end{aligned} \tag{2.56}$$

このように，掃引戦略図の各境界とそのパラメータ A〜D は求まる．時間と掃引率のモデル，GBM-based アルゴリズムの各動作制御の遷移する確率を用いて，掃引戦略図を導き出せることを示した．

2.9　小括

本章では，掃引に関する動作制御手法の提案を行った．未知環境の掃引では，掃引の完全性や効率を考えるとオンラインでの地図生成が好ましい．オンラインで生成される地図は，掃引初期から掃引終盤にかけて，徐々に信頼性を高めながら完成する．そのため，掃引終盤で効率が良い地図全体を利用した動作計画は，掃引初期には無駄になることが多い．また，ロボットは移動することで自己位置の確からしさが低下する．自己位置推定の結果と実際の位置が大きく違うと，掃引を行っても意味がない．そのときの状況に応じて，ロボットは動作制御の切り替えを行う必要がある．そこで，新たな掃引動作制御手法，Gradually Building Map-based（GBM-based）アルゴリズムを提案した．GBM-based という名称は，漸進的に生成される地図を基準に，ロボットが動作制御を大きく切り替えることに由来する．

GBM-based アルゴリズムは 3 つの動作制御手法と 2 つの条件分岐で構成される．3 つの動作制御はそれぞれ，地図周辺のみを利用する Local Reference-based，地図全体を利用する Global Reference-based，自己位置の確からしさを向上させる Landmark Search であり，「ロボット周辺に未掃引領域が存在するか」と「自己位置の確からしさが小さいか」という 2 つの条件によって各動作制御を選択する．2 つの条件分岐には，掃引の終了時刻や部屋環境の大きさに関する情報が入っていないにも関わらず，掃引開始から掃引終了にかけて徐々に Local Reference-based から Global Reference-based へと選択する割合が切り替わる点が，本提案アルゴリズムの大きなポイントである．地図の完成度が高まるにつれ，ロボット周辺に未掃引領域が存在しない確率が大きくなるため，環境の大きさに依存することなく，掃引初期には Local Reference-based が主に，掃引終盤では Global Reference-based が主に選択されると考えられる．また，自己位置の確からしさは時間に依存しないとすると，Landmark Search は一定の割合で選択されると考えられる．本提案手法のように，掃引が進むにつれ，動作制御の選択される割合が変化することを意図して設計されたアルゴリズムは存在しない．

本章では，このような掃引アルゴリズムを，掃除ロボットに適用することを想定したアルゴリズムを提案している．Local Reference-based，Global Reference-based，

Landmark Search の 3 つの動作制御，2 つの条件を設定することで，掃除ロボットのアルゴリズムを設計している．2 つの条件分岐には,「壁衝突時にロボット周辺に未掃引領域のエントロピーが閾値よりも大きいかどうか」,「壁衝突時に Kalman Filter の自己位置誤差を表す誤差共分散行列の最大固有値が閾値よりも大きいかどうか」を用いている．本提案アルゴリズムの有効性の確認として，Behavior-based アルゴリズムと，本提案アルゴリズムをそれぞれを用いて掃引するシミュレーションと実験を行った．30 回のシミュレーションの掃引率を平均した結果が，4 つの部屋形状全てで Behavior-based アルゴリズムよりも効率が良いことを示した．また，本提案アルゴリズムでは，掃引が進むにつれ，標準偏差が小さくなっていることが確認された．掃引開始時は，環境情報が乏しいことに加え，スタート地点がランダムであることから掃引率の偏差が大きいとみられ，環境情報が得られるにつれ，効率的な掃引ができていると考えられる．さらに実験でもシミュレーション同様に，本提案アルゴリズムの掃引率が高いことを示した．

また，本章ではロボットのセンサ誤差，移動誤差が無視できない場合の掃引に関する新たな問題について議論している．センサ誤差，移動誤差により掃引の後半では，未掃引領域は複数の未掃引領域の塊として存在することとなる．従来の掃引に関する研究では，この問題に関して述べられていない．この問題に対して，未掃引領域の塊を考慮した掃引計画手法を提案し，計算量の多い最適解法と計算量の少ない準最適解法に差がないことをシミュレーションにより示した．

第3章　不整地環境での移動能力を向上させる尻尾に関する研究

3.1　緒言

車両型移動ロボットは，移動の効率が良いものの，砂利や凹凸の多い不整地での移動は困難なことが知られている．シンプルな（移動以外の機構を持たない）車両型の移動ロボットは斜面や小さな障害物による転倒の危険性や図 3.1 のような移動不能な状況に陥る可能性が非常に高い．極限環境での探査を目的としたロボットにおいて，移動不能状態に陥ることは致命的である．他にも地図上に存在しない経路の断裂や水没なども想定され，ロボットはこれらの状況を解決可能な機構，動作制御が望まれる．しかし，想定される各状況に最適な機構を，ロボットに全て搭載することは，ロボットの重量やサイズ，コスト面から考えて不可能である．本章では，多くの状況に対応するため，生物界で様々な用途で利用される尻尾を，車両型の移動ロボットに適用することを考える．

本章では，大きく分けて 3 つについて提案する．1 点目は，形状を保存可能な柔軟尻尾を用いて大きな慣性力をロボット本体に伝達する点．2 点目は，車両型の移動ロボットに尻尾を搭載し，跳躍動作を行う点．3 点目に，跳躍後に目標位置姿勢に到達するための車両型移動ロボットの入力設計手法である．

近年，移動ロボットに尻尾を搭載し，尻尾を振ることでロボット本体に反力・反トルクを与える研究が盛んに行われている [56]-[80]．これらの研究では，尻尾として剛体のリンクとその先端に取り付けた錘を使用している．しかし，剛体 1 リンクでは用途が限られており，狭い領域での移動が困難であることが欠点として挙げられる．

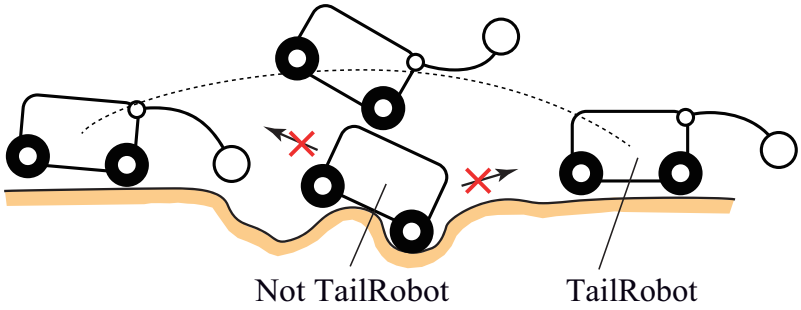


Fig. 3.1 An advantage of mobile robots with a tail.

そこで，多機能な尻尾として，可変剛性機能を有した柔軟尻尾を提案する．本尻尾では，尻尾の形状が変形するため，狭い場所での移動が可能である上に，尻尾の形状を保存することが可能である．尻尾の形状を保存するとは，尻尾の硬さを硬く変化させることで，尻尾の形状変化を抑制することを意味する．尻尾先端において押しつけ力等を発生させることが可能となり，例えばスイッチを押すなどのタスクや，ドアノブに尻尾先端を掛け，ドアを引くといったタスクにも対応が可能となる．さらに，剛体 1 リンクの尻尾に比べ，ロボット本体に大きな慣性を伝えることができるという利点を持つ．これについては，詳しくは 3.2 節で述べる．

形状保存後の尻尾を剛体とみなせば，ロボット本体と尻尾は 2 リンクのロボットと近似することができる．アクロバティックな動作を目的とする，2 リンクのロボットはアクロボット [97] とも呼ばれ，非線形制御の分野で数多く研究されており，ロボットの姿勢の制御手法を本動作制御手法に適用することができる．特に，連続的な跳躍を目的としたアクロボットの制御手法や空中浮遊時におけるアクロボットの制御手法が本研究でも重要になってくる．Berkemeier と Fearing[98] は 2 リンクのホッピングロボットの制御手法を提案している．彼らは，連続した跳躍を脚の接地した状態（Stance Phase）と浮遊した状態（Flight Phase）に分けてコントローラを設計している．Stance Phase では，2 リンク間の角度を，脚と地面間の角度に関係を持たせ，制御することで，ロボット全体を一つの振り子として考える．ロボットの運動は，振り子の 2 つのパラメータで決定される．これに，目標軌道との誤差を修正するため，フィードバック項を追加したコントローラを用いている．Flight Phase では，重力以外の外力が働かないため，ロボットの運動は角運動量保存則によって

支配される．

本章では，車両型ロボットの両輪が接地している状態を Normal Phase，前輪だけが接地している状態を Stumbling Phase，両輪が宙に浮いている状態を Flight Phase と定義する．跳躍動作を Stumbling Phase と Flight Phase の 2 つの状態に分割し，それぞれを振り子のモデルと 2 リンクの浮遊ロボットのモデルとして表現し，提案する．Stumbling Phase では，尻尾からの反力・反トルクを用い，車両型ロボットを転倒するような直立姿勢に近づける．Stumbling Phase で生じた本体の角度と角速度を用いて，宙に浮く Flight Phase に遷移する．つまり，尻尾を利用してロボットを不安定な（転倒する）状態に導いた後，尻尾を利用して安定な（転倒しない）状態に修正を行う．

図 3.1 のような場面においてロボットが跳躍を行う場合，確実にくぼみを飛び越え安全な位置にロボット本体を運ぶこと，転倒しない姿勢に制御することが重要となる．そこで，空中に浮いた目標位置姿勢を設定し，その目標位置姿勢に到達するための計画手法を提案する．3 つのモデルを用いて車両型ロボットの入力と，Phase の切り替える点の設計を行う．

本章の構成は以下の通りである．2 節では，瞬間的に形状を保存可能な柔軟尻尾について述べる．3 節では，ロボットの跳躍動作について説明し，我々が提案する 2 つの状態のモデルについて述べる．4 節では，本稿の主な提案部分である跳躍動作を行うための入力の設計法について述べる．5 節では二つの数値シミュレーションを行う．一つ目は試行錯誤的に設計した入力をロボットに与え，ロボットが跳躍可能であることを示す．二つ目は提案した入力設計法を用いて，ロボットが宙に浮いた目標位置姿勢に到達するような入力を設計を行う．この入力をロボットに与え，ロボットが目標位置姿勢に到達できることを示す．

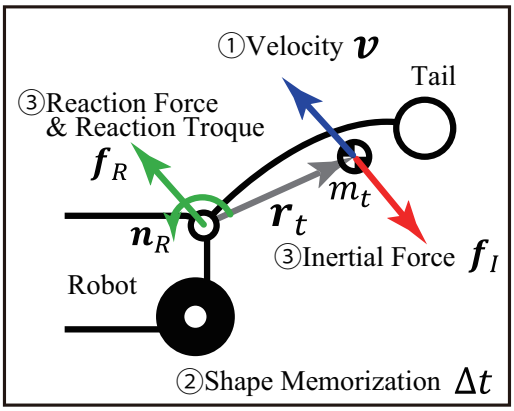


Fig. 3.2 An advantage of mobile robots with a tail.

3.2　可変剛性機能を有した柔軟尻尾

3.2.1　瞬間的に形状保存可能な尻尾の利点

剛体 1 リンクの尻尾は，根元アクチュエータの角加速度により，尻尾の重心には慣性力が発生し，ロボット本体に反力・反トルクとして伝達される．ここで，瞬間的に形状を保存可能な柔軟尻尾をロボットに搭載した場合を考える．ロボット本体に慣性力を伝達する基本的なアイデアは剛体 1 リンクの尻尾と変わらない．瞬間的に柔軟尻尾の形状を保存することに加え，尻尾根元アクチュエータの角速度をゼロにすることで，3 次元上で運動する尻尾の重心速度に疑似的なブレーキをかけ，尻尾重心に運動方向と逆向きの大きな慣性力を発生させる．これを本稿では，尻尾の形状保存（*Shape Memorization*）と呼ぶ．

今，図 3.2 青矢印のように，尻尾の重心がある速度 $\boldsymbol{v}$ で運動しているとする．尻尾の全質量を m_t とし，尻尾の重心から尻尾の根元までのベクトルを $\boldsymbol{r}_t$ と表す．このとき，尻尾の形状を保存すると，尻尾重心の速度はゼロとなり，尻尾の重心には青矢印と逆方向に慣性力が働く．この慣性力を図 3.2 赤矢印で表している．尻尾の形状保存に Δt 要するとし，この期間は加速度一定とすると，慣性力 $\boldsymbol{f}_I$ は

$$\boldsymbol{f}_I = m_t \boldsymbol{v} / \Delta t. \tag{3.1}$$

と表される．同時に慣性力は，反力・反トルクとしてロボット本体に伝達され，ロ

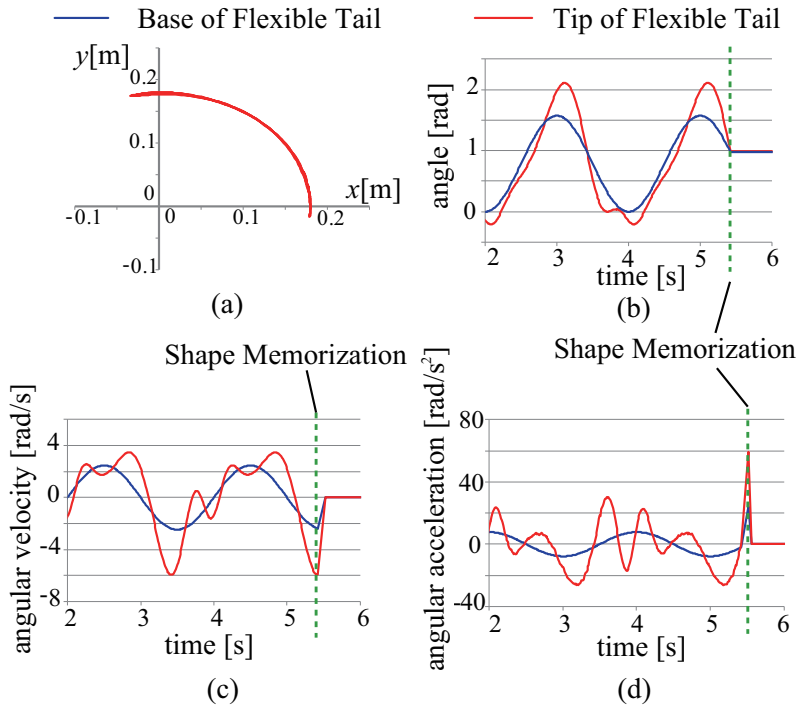


Fig. 3.3 Position, angle, angular velocity and angular acceleration of flexible tail simulation

ボット本体に運動をもたらす．反力を $\boldsymbol{f}_R$，反トルクを $\boldsymbol{n}_R$ とすると次式のように表現される．

$$\begin{aligned} \boldsymbol{f}_R &= -\boldsymbol{f}_I \\ \boldsymbol{n}_R &= \boldsymbol{r}_t \times \boldsymbol{f}_I. \end{aligned} \tag{3.2}$$

剛体 1 リンクの尻尾の重心の運動は，根元モータの角速度に必ず従うが，柔軟な尻尾の場合，根元から見た重心の相対的な角速度は根元モータの角速度と異なる．尻尾の長さ，質量等のパラメータを適切に設定すると，根元モータの角速度より大きな角速度を得られる時が存在し，根元モータの角速度が同じ場合でも，柔軟な尻尾の最大重心速度は，剛体の尻尾の最大重心速度よりも大きくなる．形状保存に要する時間 Δt が小さければ，最大重心速度時における形状保存で発生する慣性力は，剛体の尻尾を用いて根元アクチュエータの角速度をゼロにした場合以上に大きくなる．柔軟な尻尾を振るということ（柔軟物の振動）は，根元のアクチュエータの振動数に大きく影響する．以降の議論では，尻尾の振動モードが 1 次モードとなる根

元アクチュエータの入力に絞って議論を行う．

ここでロボット本体を固定して，尻尾の根元アクチュエータに $0 < \phi < \pi/2$ とする正弦波の入力を与えたときの様子を図 3.3 に示す．図 3.3(b)(c)(d) の青色の実線は根元の角度，角速度，各加速度を表している．一方，赤色の実線はそれぞれ，尻尾先端の位置と根元の位置から擬似的に 1 リンクの棒とみなした角度，角速度，各加速度を表している．この図から尻尾先端の角度，角速度ともに，尻尾先端は根元のアクチュエータに依存していないことがわかる．1 リンクの剛体尻尾の角度，角速度は必ず根元のアクチュエータの動きのみに依存するため，柔軟な尻尾は剛体尻尾に比べ自由度が大きいといえる．

図 3.3 では $t = 5.3$[s]（緑の点線の時刻）で尻尾の形状保存を行ったときの様子が確認できる．運動中の尻尾の形状保存に加え，根元の関節を停止させると大きな加速度が発生する．(図 3.3(d)）大きな角加速度を得ることができれば効率的にロボット本体の運動を変化させることが可能である．そのため，適切なタイミング設定することで，根元のアクチュエータの速度が同じであっても，剛体尻尾に比べ柔軟尻尾のほうがロボット本体へ大きな反力．反トルクを与えることが可能である．

3.2.2　形状を保存可能な柔軟尻尾の機構例

運動中に形状を保存する柔軟尻尾の例として，我々が開発している可変剛性機構を有する柔軟尻尾を図 3.4 に示す．本尻尾は柔軟な線材（Ni-Ti）が中心に位置し，長手方向に骨と呼ばれる剛体が等間隔に取り付けられている．柔軟な線材の先端には錘が取り付けられており，尻尾の重量は主に先端に集中している．一つの剛体骨は，上下の剛体骨と可変剛性機構で接続されており，可変剛性機構の剛性が高くなることで，その時の 2 つの剛体骨の角度が維持され，2 つの剛体骨間の柔軟な線材の曲率は保存される．

可変剛性機構は，袋とその中の粒子（砂）で構成されている．本可変剛性機構は，ジャミング効果 [99] として知られている現象を利用している．ジャミング効果は袋内の圧力を，大気圧から負圧へと切り替えることで発生する．袋内が大気圧の状態では粒子は袋内で流動し，袋外部からの力に受動的に変形する．袋内が負圧になる

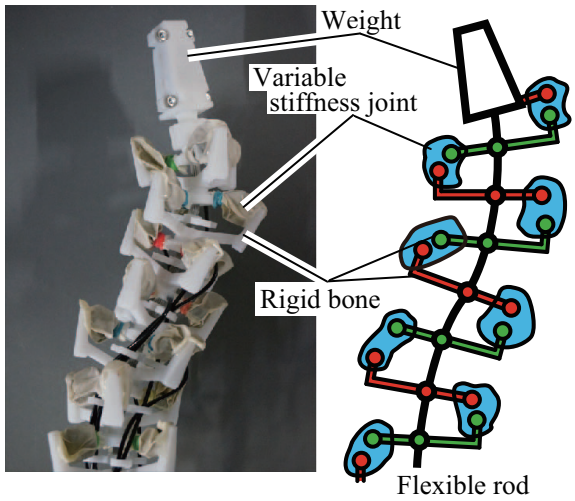


Fig. 3.4 Our flexible tail equipped with shape memorization mechanism.

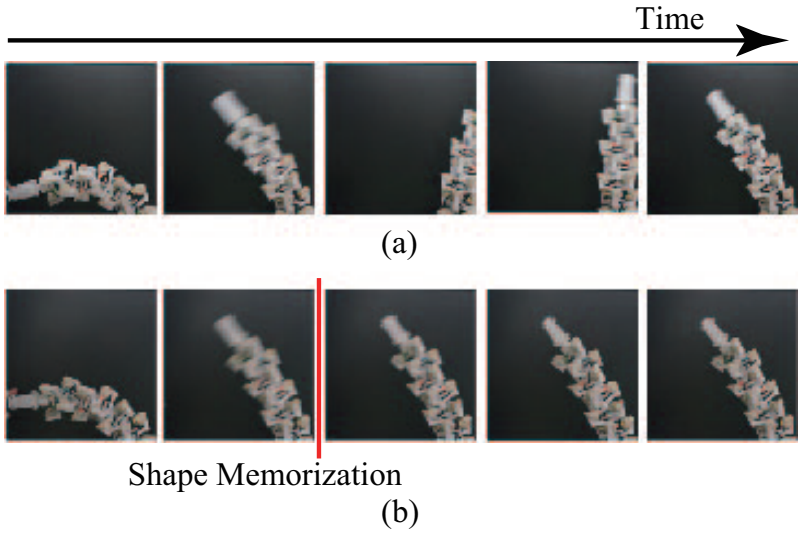


Fig. 3.5 Snapshots of Shape Memorization of our tail.

と，粒子は圧縮され，粒子間もしくは粒子と袋間で摩擦が発生するため，機構が非常に硬くなる．類似した機構として，Kim らにより開発された，シート間の摩擦を利用する Layer Jamming が挙げられる [100]．複数の可変剛性機構を有する尻尾は，すべての可変剛性機構を個別に制御することで，非常に複雑な尻尾の制御が可能となるが，制御装置もその数だけ必要となるため，全体の機構が大掛かりなものとなる．そのため，すべての可変剛性機構を一つの制御装置を用いて剛性を変化させる．

つまり，すべての可変剛性機構が同時に硬い状態から軟らかい状態へ，または軟らかい状態から硬い状態へ変化する．すべての可変剛性機構が同時に硬い状態から軟らかい状態へ変化することで，尻尾の形状は保存される．図 3.5(a) では本尻尾を柔軟な状態で運動させ続けたときの様子を示している．可変剛性機構の剛性が低い場合には，尻尾は柔軟であり，減衰振動する．一方，図 3.5(b) は本尻尾の形状を尻尾の運動中に保存した時の様子を示している．尻尾すべての可変剛性機構を軟らかい状態から硬い状態に切り替えると，それ以降は尻尾の形状が保存されていることが写真から見て取れる．

3.3　尻尾搭載型車両ロボットの跳躍動作

3.3.1　跳躍動作計画問題

図 3.1 のような場面においてロボットが跳躍を行う場合，確実にくぼみを飛び越え安全な位置にロボット本体を運ぶことが重要となる．また，跳躍が成功したとしても，ロボットの姿勢によっては着地時に転倒する可能性がある．そのため，我々は図 3.6 に示すような，尻尾を搭載した車両型移動ロボットの動作計画手順を考える．

類似した研究，2 リンクのロボットのための跳躍に関する制御手法として，Berkemeier ら [98] によって考案された手法が挙げられる．これは，2 段階に分かれており，接地時のバランス制御と，浮遊時の姿勢制御に分かれている．しかし，4 輪車両ロボットは連続的な跳躍は必要なく，むしろ目標位置姿勢に到達することを目的とする．

このような問題を考えるため，ロボットの両輪が接地している状態を *Normal Phase*，両輪とも地面から離れている状態を *Flight Phase* と定義する．Normal Phase から Flight Phase に遷移することを考えると，ロボット本体に上方向の大きな加速度が必要となる．今，ロボットの持つ入力は尻尾根元のモータ，形状保存機能，車輪の 3 つであることを考慮すると，上方向の大きな加速度を得るためには，尻尾による大きな反力・反トルクを利用することが最も有効である．尻尾先端の錘を重くする，または尻尾長を長く設定することで，尻尾による大きな反力・反トルクは得られる

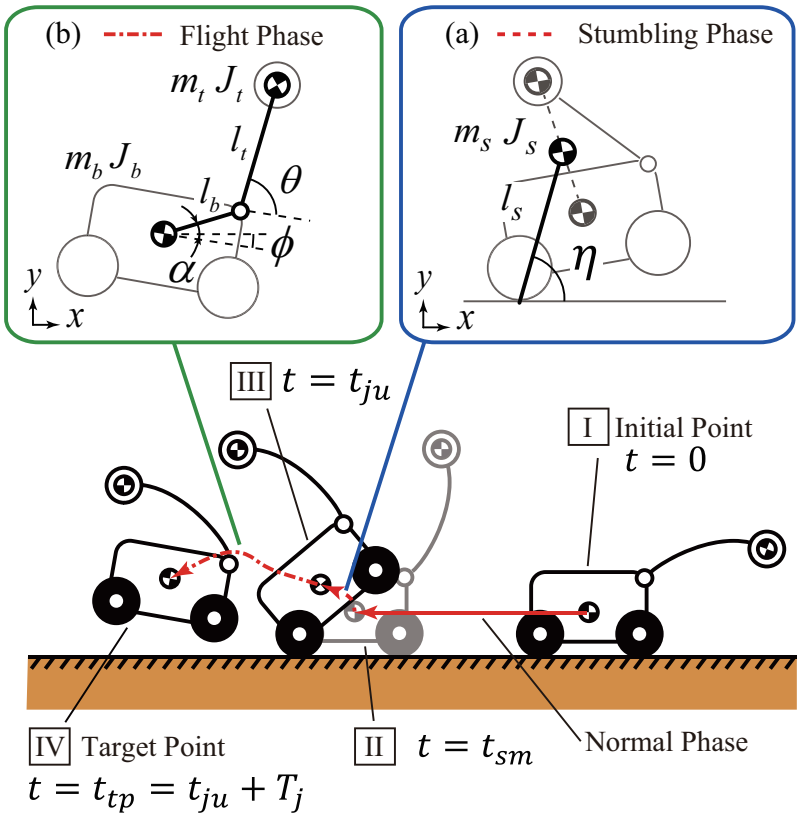


Fig. 3.6 The schematic of the motion planning problem and two phases which are divided a jump motion, Stumbling Phase and Flight Phase.

が，尻尾が移動ロボットの移動を妨げとなるような質量や長さは好ましくない．そこで，タイヤの入力と尻尾の運動を用い，図 3.6(a) に示すロボット本体が躓くような運動を誘導する．このような，前輪のみが接地した状態を *Stumbling Phase* と定義する．ロボット本体がある程度大きな傾き角を得た後に，尻尾を用いてロボット本体の角度を制御し浮遊させる．

跳躍の動作計画問題には 3 つの状態が存在するので，各状態の入力と，2 つの状態切り替え点を設計することが必要である．形状保存のタイミングを t_{sm}，前輪が地面から離れるタイミングを t_{ju} と定義する．形状保存のタイミング t_{sm} では，その直前と直後でエネルギー保存則が成立する．Stumbling Phase におけるロボットの躓くような運動では，ロボット全体重心-水平面間の角度と角速度は，前輪接地点を

中心点とした振り子の運動に従う．Flight Phase ではロボットに重力以外の外力が働かないため，尻尾を含めたロボット全体の重心の並進方向の運動は，前輪が地面から離れるタイミング t_{ju} 以降，放物運動に従い，ロボットの姿勢は角運動量保存則に従う．このような拘束を考慮し，t_{sm}，t_{ju}，Flight Phase でのコントローラを設計し，目標値まで制御する問題となる．

以下の節では，各 Phase のロボットのモデルについて説明する．

3.3.2　Normal Phase

Normal Phase は両輪が接地した状態であり，一般的な車両型ロボットの状態のことを呼ぶ．ロボット本体が尻尾よりある方向に大きな反力・反トルクを受けると，タイヤ前輪と地面との接触位置を中心として傾き，躓くような動作を引き起こす．本稿では，尻尾の形状保存時に，Stumbling Phase に移行するよう，尻尾を制御する．

Normal Phase 時に助走を行うことで，前輪の接地点を中心とした初期角速度 $\dot{\eta}_0$ は大きなもとのなる．$\dot{\eta}_0$ は，形状保存時や地面と前輪間の摩擦による損失がなければ，形状保存前後のエネルギー保存則により，次式から求められる．

$$\frac{1}{2}m_b\dot{x}^2 + \frac{1}{2}m_t|\boldsymbol{v}_t|^2 + \frac{1}{2}J_t\dot{\theta}^2 = \frac{1}{2}(J_s + m_s l_s^2)\dot{\eta}_0^2, \tag{3.3}$$

$\dot{x}$ は，ロボットの水平方向速度を表している．m_s, l_s, J_s は，図 3.6（a）に示す通り，振り子の幾何パラメータ（質量，振り子長，慣性モーメント）である．

動作計画問題を解くときには，あらかじめ用意された尻尾根元モータの軌道 $\theta_{np}(t)$ を用いて計画を行う．形状保存のタイミングがある時刻 t_{sm} と決まったとき，Normal Phase 時（$t < t_{sm}$）における尻尾根元モータ入力 τ_m とタイヤ入力 τ_w は

$$\begin{aligned}\tau_m &= K_p^t(\theta_{np}(t) - \theta_1) + K_d^t(\dot{\theta}_{np}(t) - \dot{\theta}_1)\\ \tau_w &= K_{p,1}^w(\dot{x}_{np}^w - \dot{x}^w),\end{aligned} \tag{3.4}$$

ここで，$\theta_1, \dot{\theta}_1$ は根元モータの角度・角速度，$\dot{x}^w$ は前輪の水平方向速度を意味している．また，$\dot{x}_{np}^w$ は目標速度，$K_{p,1}^t, K_{d,1}^t, K_{p,1}^w$ は定数ゲインを表している．

3.3.3 Stumbling Phase

Stumbling Phase は，前輪のみが接地している状態を表し，2 リンクホッピングロボットにおける Stance Phase[98] と似た状態である．Stance Phase はロボットのバランス制御を行うのに対し，Stumbling Phase ではバランス制御を行わない．Stumbling Phase では，$\dot{\eta}_0$ の勢いに任せて，ロボット本体を傾ける．

形状保存のタイミング t_{sm} 以降，柔軟尻尾は形状が維持されたままとすると，剛体の 1 リンクの尻尾とみなすことができる．さらに，ロボット本体と尻尾の位置関係を崩さなければ，ロボット全体は前輪の接地点を中心とした振り子のモデルとして表現できる（図 3.6（a））．この時刻の尻尾根元アクチュエータの角度を θ_{sm} として表す．エネルギー保存則より，振り子の角度 η と角速度 $\dot{\eta}$ 間の関係式は以下の式で表される．

$$\frac{1}{2}\dot{\eta}^2 = A_p \sin\eta + \frac{1}{2}\dot{\eta_0}^2 - A_p \sin\eta_0, \tag{3.5}$$

A_p は

$$A_p = -\frac{m_s g l_s}{J_s + m_s l_s^2}. \tag{3.6}$$

Stumbling Phase 中の振り子の幾何パラメータと，振り子回転中心を固定することで，ロボット全体の運動の予測が容易となる．それ故に Stumbling Phase では，尻尾根元モータの角速度を 0 にするように制御する．また，振り子の回転中心を移動させないため，前輪の位置の水平方向速度 $\dot{x}_{ft}$ を 0 にするよう制御を行う．離陸のタイミング t_{ju} を決定すると，Stumbling Phase 時（$t_{sm} \leq t < t_{ju}$）の尻尾根元モータの入力 τ_m とタイヤ入力 τ_w は，以下の式で表現される．

$$\begin{aligned} \tau_m &= K^m_{p,2}(\theta_{sm} - \theta_1) - K^m_{d,2}\dot{\theta}_1 \\ \tau_w &= -K^w_{p,2}\dot{x}^w, \end{aligned} \tag{3.7}$$

$K^t_{p,2}, K^t_{d,2}, K^w_{p,2}$ は定数ゲインである．

3.3.4 Flight Phase

Flight Phase でも尻尾は保存された形状を維持したままだとすると，ロボットは剛体 2 リンクのロボットとみなすことができる．浮遊時，ロボットには重力を除く

外力は働かないので，剛体 2 リンクのロボット全体の重心位置は放物運動する．そのため，ロボットが地面から離れる直前の重心の位置，速度が決定すると，目標の重心位置に到達するまでの時間を求めることができる．ロボットが地面から離れるタイミングを $t = t_{ju}$(図 3.6(III)) とする．T_j を図 3.6(III) から（IV) までの跳躍時間と定義し，$\dot{\tilde{x}}_j, \dot{\tilde{y}}_j$ を図 3.6(III) でのロボット全体重心の水平方向速度，鉛直方向速度とする．このとき，図 3.6(III) から（IV) までの全体重心の位置に関する拘束は，

$$\begin{aligned} \tilde{x}_r - l_s \cos\eta_j &= \dot{\tilde{x}}_j T_j \\ \tilde{y}_r - l_s \sin\eta_j &= \dot{\tilde{y}}_j T_j - \frac{1}{2} g T_j^2. \end{aligned} \tag{3.8}$$

となる．

さらに，重力以外の外力が加わっていない，空中に浮遊している物体の角運動量が保存される．この保存則を利用して，2 リンクのロボットは姿勢を修正することができる．浮遊している初期角運動量が非ゼロである 2 リンクの角運動量保存式は非ホロノミック拘束であることが知られており [97]，多くの姿勢の制御手法が提案されている．この問題は，ある決められた時間の間に，2 リンクの浮遊ロボットが目標の姿勢まで制御できるかという問題に置き換えることができ，このような問題を解決する制御手法として，Xin らにより提案された手法 [101] と服部らにより提案された手法 [102][103] が挙げられる．両手法とも Back Stepping 法を用いて，コントローラを設計しているが，両者はロボットのモデルに大きな違いがある．Xin らは入力として 2 リンク間の角加速度とし，服部らはトルクを入力としてモデルをたてている．我々は実機での適用を踏まえ，服部らの手法を Flight Phase 時の制御手法として採用する．角運動量保存則の式は，初期角運動量を P_0 とすると，

$$P_0 = (M_1 + A_1 \cos(\theta - \alpha))\dot{\phi} + (M_2 + A_2 \cos(\theta - \alpha))\dot{\theta} \tag{3.9}$$

ロボット本体と地面間の相対角を ϕ，尻尾とロボット本体との相対角度を θ とする．α は図 3.6 (b) で示すように，ロボット本体の重心までのオフセット角を表している．M_1, M_2, A_1, A_2 は幾何パラメータ $m_b, m_s, J_b, J_s, l_b, l_s$ で表現される係数である．以下では，初期角運動量が非ゼロの浮遊 2 リンクロボットを，設定された時刻に目標姿勢へと到達させるための制御手法 (服部らによって提案された)[102][103] について述

べる．この方法を利用することで，ロボットは位置と姿勢を同時に目標位置姿勢へと到達させることが可能となる．新たな変数 ψ を定義する．

$$\psi = \phi - \phi_r + \int_{\theta_r}^{\theta} \frac{M_2 + A_2 \cos\lambda}{M_1 + A_1 \cos\lambda} d\lambda \tag{3.10}$$

$$\psi_r = \frac{P_0}{M_1 + A_1 \cos\psi}(t - T), \tag{3.11}$$

とおき，以下のように状態変数を目標との誤差で定義する．

$$\begin{aligned} x_1 &:= \psi - \psi_r \\ x_2 &:= \phi - \phi_r \\ x_3 &:= \dot{\phi}. \end{aligned} \tag{3.12}$$

このとき，尻尾根元アクチュエータの入力をトルクとした運動方程式から，システムの状態方程式は次のようになる．

$$\begin{bmatrix} \dot{x_1} \\ \dot{x_2} \\ \dot{x_3} \end{bmatrix} = \begin{bmatrix} f_1(x_2) \\ x_3 \\ f_3(x_2, x_3) \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ g_3(x_2) \end{bmatrix} u. \tag{3.13}$$

ここで用いられる関数 f_1, f_3, g_3 は，以下の式の通りである．

$$\begin{aligned} f_1(x_2) &= -\frac{P_0}{K_1} + D \\ f_3(x_2, x_3) &= \frac{A_2 K_3}{K_0 K_1} \sin(x_2 + \psi_f) \\ g_3(x_2) &= \frac{K_1}{K_0} \\ D &= \frac{P_0}{M_1 - A_2 \cos\psi_f} \\ K_0(x_2) &= M_2(M_1 - M_2) - A_2^2 \cos^2(x_2 + \psi_f) \\ K_1(x_2) &= M_1 + A_1 \cos(x_2 + \psi_f) \\ K_2(x_2) &= M_2 + A_2 \cos(x_2 + \psi_f) \\ K_3(x_2, x_3) &= (2M_2 - M1 - K_2) K_2 x_3^2 - P_0^2 \end{aligned} \tag{3.14}$$

服部らの提案手法 [102][103] では，根元アクチュエータの入力 τ_m は次のように表現される．

$$\tau_m = \frac{P^* - f_3(x_2, x_3)}{g_3(x_2)}, \tag{3.15}$$

P^* は，

$$P^* = -(k_2 k_3 + \dot{k_2})(x_2 - Q) - (k_2 + k_3)(x_3 - \dot{Q}) + \ddot{Q} \tag{3.16}$$

$$Q = \begin{cases} -\psi_f - \cos^{-1}\Omega & (-\pi < \psi_f < 0) \\ -\psi_f + \cos^{-1}\Omega & (0 < \psi_f < \pi) \end{cases} \tag{3.17}$$

$$\Omega = \frac{P_0}{A_1(\xi(x_1, k_1) + D)} - \frac{M_1}{A_1} \tag{3.18}$$

Flight Phase 時（$t_{ju} \leq t \leq t_{ju} + T_j$），尻尾根元モータの入力 τ_m にこの式を用いる．浮遊時には，前輪の入力 τ_w は意味をなさないため，ゼロと設定する．

$$\tau_w = 0 \tag{3.19}$$

初期角運動量が非ゼロである浮遊ロボットは可到達なシステムであり，制御時間は初期角運動量により制限されることにも注意しなければならない．制御時間は，次式によって制限される [97].

$$-\frac{M_1 - A_1}{P_0}\psi(0) \leq T_j \leq -\frac{M_1 + A_1}{P_0}\psi(0), \tag{3.20}$$

M_1, A_1 は式（3.9）と同じパラメータである．浮遊時間 T_j が，式（3.20）を満たせば，状態フィードバックコントローラを用いてロボットの姿勢を目標姿勢まで制御することができる．

3.4　目標位置姿勢に到達するための入力設計手法

この節では，初期位置から空中に浮いた目標位置姿勢に到達するための入力を計画する手法について述べる．まず，尻尾の形状保存を行う時刻 t_{sm} を決定する，この時，あらかじめ用意した尻尾根元モータの軌道 $\theta_{np}(t)$（$t \leq t_{sm}$）としてシミュレーションを行うことで，形状保存直後の振り子の幾何パラメータ A_p，初期角度 η_0，初

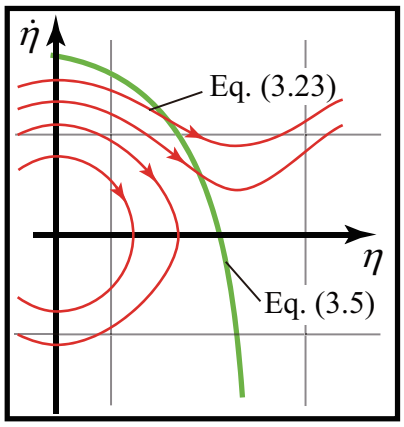


Fig. 3.7 The graph illustrates Eq. 3.5 (red trajectory) and Eq. 3.23 (green curve). An intersection between a red line and a green line is a solution of which the robot's COM can reaches the target position.

期角速度 $\dot{\eta}_0$ を得る．次に，A_p，η_0，$\dot{\eta}_0$ を式（3.5）に代入することで，振り子の角度と角速度の関係を求める．この関係により，図 3.7 での一つの赤色の線が決定することを意味する．

図 3.6 (a) での振り子の角度 η と角速度 $\dot{\eta}$ が与えられた時，ロボットの重心の水平方向と垂直方向の速度 $\dot{\tilde{x}}_j, \dot{\tilde{y}}_j$ は，それぞれ以下のように与えられる．

$$\begin{aligned} \dot{\tilde{x}}_j &= -l_s \dot{\eta}_j \sin \eta_j \\ \dot{\tilde{y}}_j &= l_s \dot{\eta}_j \cos \eta_j. \end{aligned} \tag{3.21}$$

浮遊時には，尻尾を含めたロボット全体の重心速度は Stumbling Phase と Flight Phase の切り替え時 (図 3.6(III)) の全体の重心速度 $\dot{\tilde{x}}_j, \dot{\tilde{y}}_j$ と等しくなる．図 3.6(III) から（IV) までの跳躍時間を T_j とすると，全体の重心の目標位置 $\tilde{x}_r, \tilde{y}_r$ と跳躍時間 T_j の関係は次の式で表される．

$$\begin{aligned} \tilde{x}_r - l_s \cos(\eta_j + \gamma) &= \dot{\tilde{x}}_j T_j \\ \tilde{y}_r - l_s \sin(\eta_j + \gamma) &= \dot{\tilde{y}}_j T_j - \frac{1}{2} g T_j^2. \end{aligned} \tag{3.22}$$

尻尾を含めたロボット全体の重心の目標位置 $\tilde{x_r}, \tilde{y_r}$ はロボット本体の重心目標位置・姿勢 x_r, y_r, ϕ_r と尻尾根元角度 θ_r と尻尾の形状により計算される．尻尾の形状保

存以降，尻尾の形状は変化せず剛体とみなしているため，$\tilde{x}_r, \tilde{y}_r$ は容易に求めることができる．

式（3.21）を式（3.22）に代入し，T_j を消去することで以下の式が得られる．

$$\begin{aligned} X &= \tilde{x}_r + l_s \cos\eta_j \\ Y &= \tilde{y}_r - l_s \sin\eta_j \\ l_s^2\dot{\eta}_j^2 Y \sin^2\eta_j + l_s^2\dot{\eta}_j^2 X \cos\eta_j \sin\eta_j + \frac{1}{2}gX^2 &= 0, \end{aligned} \tag{3.23}$$

X, Y は跳躍する直前の重心から目標位置姿勢の重心までの距離を表す．この式は浮遊時の位置に関する拘束を表しており，Stumbling Phase におけるロボットの角度-角速度間の空間で描画すると，図 3.7 の緑線のようになる．緑線上の 1 点が決定すると，式（3.22）と式（3.21）から目標位置までの時間 T_j は以下のように求められる．

$$T_j = \frac{\tilde{x}_r + l_s \cos\eta_j}{l_s\dot{\eta}_j \sin\eta_j}. \tag{3.24}$$

式（3.5）と式（3.23）が交差する領域を U_s と表すと，Stumbling Phase と Flight Phase の切り替え点は U_s 上に存在する．この緑線と赤線が交わる点が，ロボット全体の重心が目標位置に到達可能な解となるが，姿勢に関して制御可能かどうかは，跳躍直前の姿勢，角運動量や目標姿勢に依存する．姿勢に関して制御できるかは，浮遊時間 T_j が式（3.20）を満たすかどうかで判定が可能である．この式を満たすような T_j となる緑と赤線の交点が最終的な解となる．

まとめると，式（3.5）と式（3.23）の交点に対して式（3.20）を満たすか判別していく．T_j が式（3.20）を満たし適切にゲイン k_1, k_2, k_3[102] が設定されたとき，式（3.15）の姿勢修正のための入力 u が求まる．

3.5　柔軟尻尾を用いた跳躍シミュレーション

本節では，柔軟尻尾を用いて車両型移動ロボットが跳躍可能であること，提案する入力設計手法の有効性を確認するため，シミュレーションを行う．まず，柔軟尻尾を n 個のリンクにて近似し，尻尾とロボット本体を含めて運動方程式を導出を行う．その後，試行錯誤的に設計した入力，入力設計手法により得た入力をそれぞれ運動方程式に与え，その結果を示す．

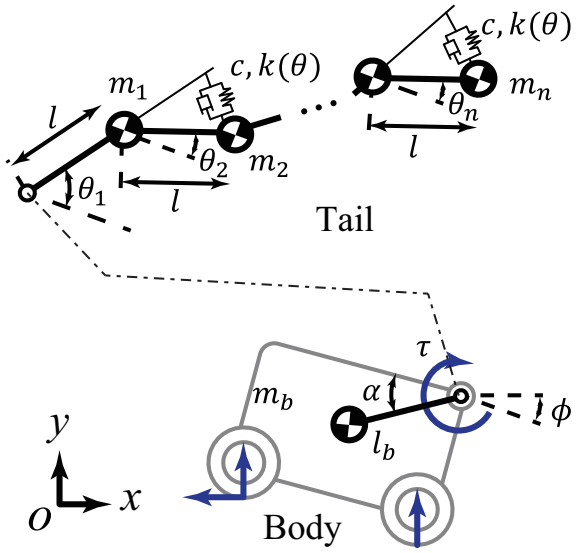


Fig. 3.8 The model of a mobile robot equipped with a flexible tail approximated by multiple links for jumping simulation.

3.5.1　柔軟尻尾を含む車両型移動ロボットの運動方程式

柔軟尻尾を n 個のリンクにて近似し，運動方程式を導出する．尻尾とロボット本体のパラメータを図 3.8 に示している．尻尾根元位置を x，y で表し，ϕ はロボット本体の姿勢を表している．リンク i は尻尾の i 番目のリンクであり，リンク i とロボット本体との相対角度を θ_i として表現している．リンク長 l，弾性係数 k，粘性係数 c はすべて同じ値としている．一般化座標を $\boldsymbol{q} = (x, y, \phi, \theta_1, \cdots, \theta_n)^T$ のようにとると，運動方程式はラグランジュ法より，

$$\boldsymbol{M}\ddot{\boldsymbol{q}} + \boldsymbol{c}(\dot{\boldsymbol{q}}, \boldsymbol{q}) + \boldsymbol{h}(\boldsymbol{q}) + \boldsymbol{k}(\boldsymbol{q}) = \boldsymbol{F} - \boldsymbol{d}(\dot{\boldsymbol{q}}), \tag{3.25}$$

$\boldsymbol{M}$ は慣性行列，$\boldsymbol{c}$ は遠心力の項，$\boldsymbol{h}$ は重力項，$\boldsymbol{k}$，$\boldsymbol{d}$ は尻尾の弾性，粘性項を表している．外力 $\boldsymbol{F}$ は，根元関節トルク τ_1 と前輪の速度 u_w と前輪・後輪に加わる抗力・摩擦力 $\boldsymbol{f}_{g1}, \boldsymbol{f}_{g2}$ から構成され，前輪・後輪から尻尾根元へと変換する座標変換行列を $\boldsymbol{J_1}, \boldsymbol{J_2}$ とおくと

$$\begin{aligned} \boldsymbol{F} &= [\boldsymbol{F}_{body}\ \boldsymbol{\tau}]^T \\ \boldsymbol{F}_{body} &= \boldsymbol{J_1}(\boldsymbol{f_{g1}} + \tau_w r_w) + \boldsymbol{J_2 f_{g2}} \end{aligned} \tag{3.26}$$

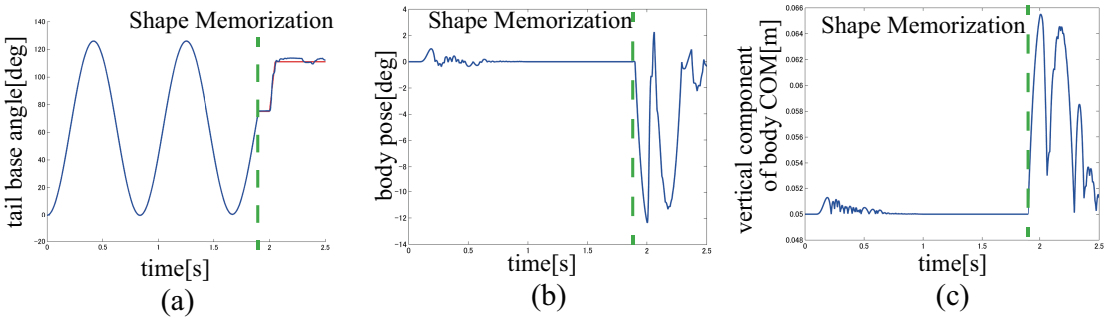


Fig. 3.9 (a) The input that we managed to design without the planning method. (b) The result of the robot body pose. (c) The result of the vertical component of the body COM position.

トルク $\boldsymbol{\tau}$ は，尻尾のトルク τ_m を用いて，$\boldsymbol{\tau} = (\tau_m, 0, \cdots, 0)^T \in \mathbb{R}^n$．$r_w$ はタイヤの半径を表している．摩擦力と接触力に関するモデルは，Xiong らにより提案されたモデル [104] を用いている．形状保存後，尻尾のモデルは n 個の剛体リンクから，尻尾の重心を基準とした仮想的な 1 個の剛体リンクへと変換している．

3.5.2　適当な入力を与えたときの跳躍動作

柔軟尻尾を用いてロボット本体が跳躍動作を行うことができるか確認するため，シミュレーションを行う．尻尾根元アクチュエータの入力，状態の切り替え点は試行錯誤的に決定したものを式（3.25）に与える（図 3.9 (a)）．ロボットの前輪には地面との接触位置での速度を 0 とするよう入力を与えており，地面と接触していないときにはタイヤの入力は働かないこととする．本シミュレーションでは，タイヤが地面から受ける抗力はバネ・ダンパモデルを用いた．

図 3.9 (b) と（c）にロボット本体の角度と重心の垂直位置をグラフを，図 3.10 はシミュレーションによるロボット本体と尻尾の挙動を表している．尻尾の形状保存が行われた $t = 1.90$[s] より，ロボット本体は前輪と地面との接地点を中心として，振り子に類似した運動をしていることが見て取れる．ロボットが形状を保存したまま，尻尾根元アクチュエータを動かすと，前輪が地面より離れ，ロボット本体が浮遊していることがわかる．非常に微小ではあるが，柔軟尻尾を用いてロボットが跳

躍できることを示した．

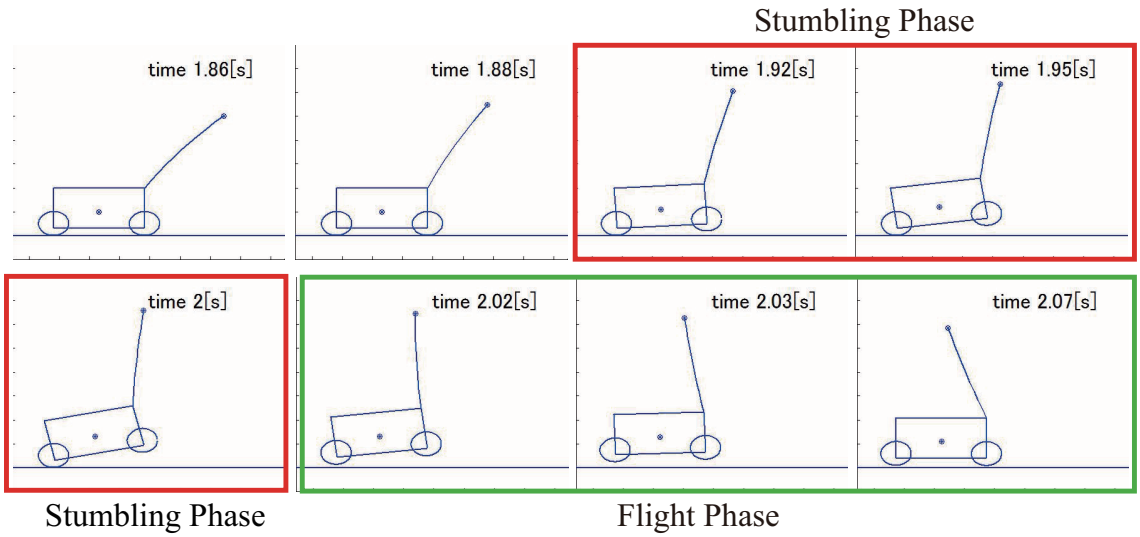


Fig. 3.10 Snapshot of the robot's jumping simulation.

3.5.3　提案する入力設計手法の有効性の確認

入力の設計

プランニングとシミュレーションに用いたパラメータを表 3.1 に示す．また，プランニングに用いた初期位置 x_0, y_0 と目標位置姿勢 x_r, y_r, ϕ_r を表 3.2 に示す．図 3.11 の尻尾根元モータの角速度を用いて，$1.38 < t < 1.56$[s] までの間で形状保存を行うタイミング t_{sm} をとるよう計画した．プランニングの結果，形状保存の時間 t_{sm}，離陸の時間 t_{ju}，目標までの到達時間 T_j は $t_{sm} = 1.511$[s], $t_{ju} = 1.558$[s], $T_j = 0.023$[s].

シミュレーション結果

シミュレーション結果のグラフを図 3.14 に示す．それぞれ，水平方向位置 x, 垂直方向位置 y, 姿勢 ϕ を表している．オレンジ，青の縦線は形状保存するタイミング t_{sm}，離陸するタイミング t_{ju} を表しており，緑の縦線は，目標位置姿勢に到達するタイミング $t_{ju} + T_j$ を表している．実線はそれぞれのパラメータの変化を，点線は

Table 3.1 Robot Parameters for the Planning and the Simulation

number of rigid link n	10
robot body mass m_b	1.0 [kg]
length from tail base to body's COM l_b	0.091 [m]
robot's body inertia J_b	0.027 [kg · m^2]
offset angle $alpha$	6.43 [deg]
weight of a rigid link m_i $(i \neq n)$	0.023[m]
tail tip weight m_i $(i = n)$	0.1[kg]
length of a rigid link l	0.02[m]
viscosity between two links c	0.001
elasticity between two links k	1.15

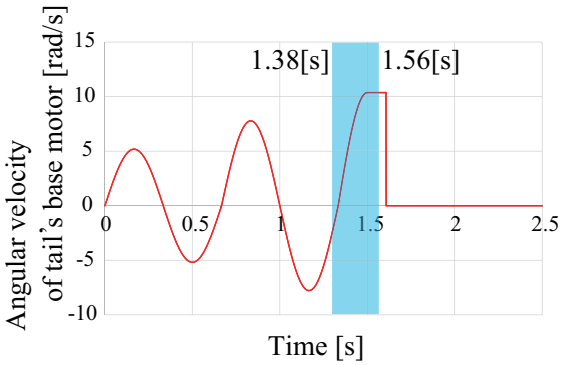


Fig. 3.11 The input used in the planning of Sec. 3.5.3

目標値を示している．x, y, ϕ すべてが，$t_{tp} = t_{ju} + T_j$ にて，目標値に近づいていることがわかる．図 3.14 より，t_{tp} は両輪が地面から離れている期間に含まれていることから，ロボットは空中に浮いた目標位置姿勢に到達できたことがわかる．

浮遊時間 T_j は非常に小さく，この短い期間で姿勢の修正を行う必要があり，実現可能には程遠い．また，水平方向の移動量は小さく，地球の重力下での実用には非常に難しいように考えられる．しかし，地球よりも重力の小さな環境では，ロボットの鉛直方向落下時間が長くなるため，水平方向の移動量は大きくなる．これに伴い，浮遊時間も大きくなることから，実現可能な入力が得られると考えられる．

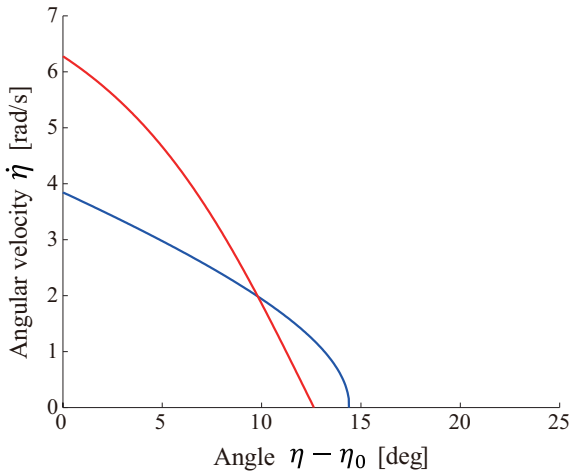


Fig. 3.12 The result of planning is illustrated. The red curve means Eq. 3.5, and the green curve means Eq. 3.23. An intersection between a red line and a green line is a solution of which the robot's COM can reaches the target position.

Table 3.2 Initial Position and Target Point for the Planning

initial position x_0	0.30 [m]
initial position y_0	0.10 [m]
initial attitude ϕ_0	0.0 [deg]
initial tail angle θ_i	30 [deg]
target position x_r	0.29 [m]
target position y_r	0.11 [m]
target attitude ϕ_r	0.0 [deg]
target tail angle θ_r	120 [deg]

3.6　小括

本章では，不整地環境にて車両型移動ロボットの移動能力を高めるための一つの動作制御手法について提案した．本章での主張は次の 3 点である．1 点目は，瞬間的に形状を保存可能な柔軟尻尾を用いて，大きな慣性力をロボット本体に伝達する点．2 点目は，尻尾を搭載した車両型の移動ロボットの跳躍動作モデル．3 点目は，尻尾を利用した車両型ロボットの，宙に浮いた目標位置姿勢に到達するための入力設計手法である．これは車両型移動ロボットに尻尾を搭載することで，これまで接

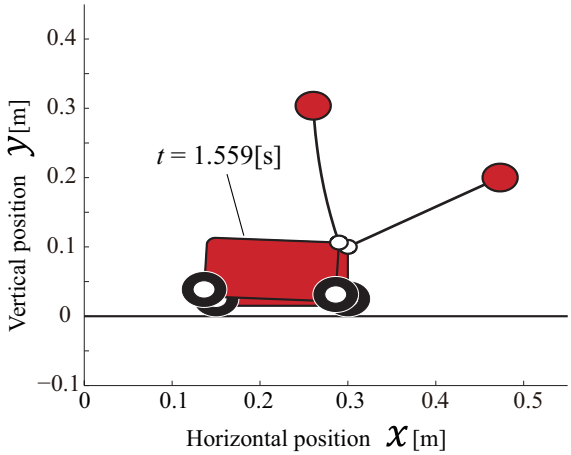


Fig. 3.13 The initial position and attitude of the robot and the target position and attitude of Simulation 2 are illustrated.

地状態でしか考えられてこなかった運動を，浮遊した状態にまで拡張するものである．本章での提案手法は図 3.1 のように車両型のロボットが不整地で移動不能状態になることを未然に回避する，または移動不能状態から脱する手段につながる．

瞬間的に柔軟な尻尾を保存することができると仮定したとき，運動している尻尾重心は瞬間的に柔軟尻尾の形状を保存することに加え，尻尾根元アクチュエータの角速度をゼロにすることで，3 次元上で運動する尻尾の重心速度に疑似的なブレーキをかけ，尻尾重心に運動方向と逆向きの大きな慣性力を発生させる．

跳躍動作は，ロボットの転倒を誘発するような状態（Stumbling Phase）と，転倒しないようロボットの姿勢を修正することで浮遊する状態（Flight Phase）の 2 つのモデルに分けられる．跳躍の動作計画問題には 3 つの状態が存在するので，3 つの状態の入力と，2 つの状態切り替え点を設計することが必要である．振り子に近似された状態（Stumbling Phase）では，尻尾の根元のモータは角度を一定に制御し，空中浮遊状態（Flight Phase）には，服部らの状態フィードバックコントローラ [103] を用いる．

本論文では，数値シミュレーションにより 2 つのことを確認している．一つ目のシミュレーションでは，適当な入力をロボットに与えることで，ロボットが跳躍可能であることを示している．2 つ目のシミュレーションにて，提案する入力設計手

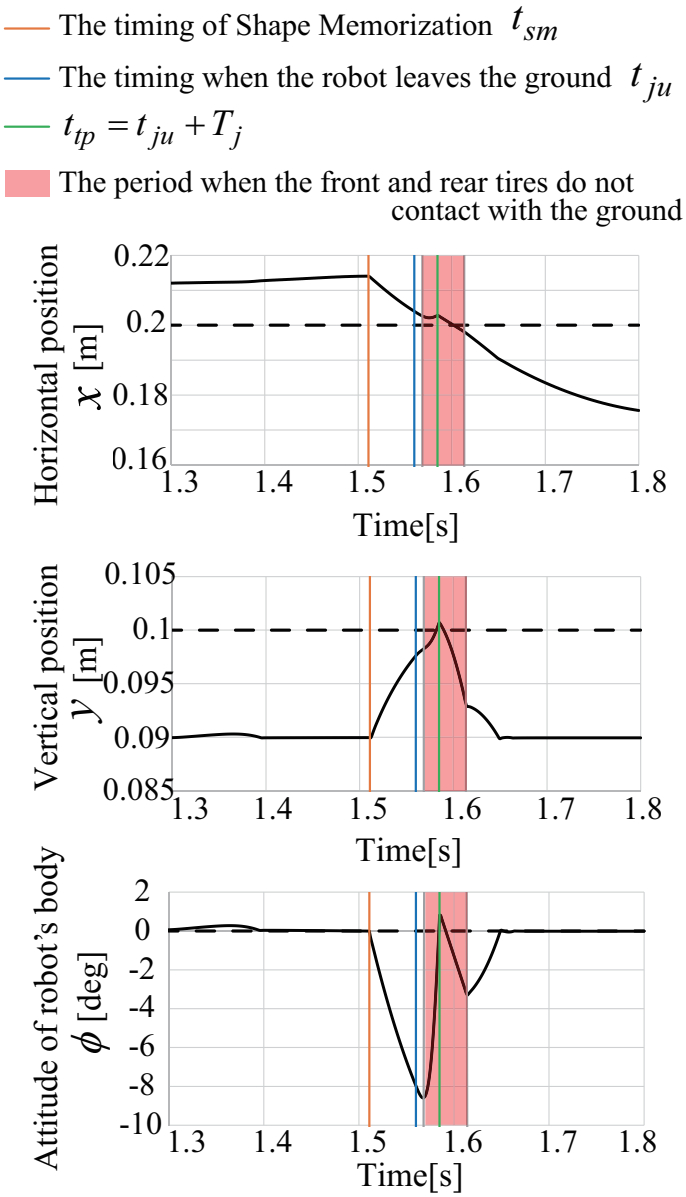


Fig. 3.14 the result of the horizontal position x, the vertical position y and the attitude of the robot's body ϕ in simulation is shown.

法を用いて設計した入力が，宙に浮いた目標位置姿勢に到達することを確認した．

2 つ目のシミュレーションで得られた浮遊時間 T_j は非常に小さく，この短い期間で姿勢の修正を行う必要があり，実現可能とは言えない．また，水平方向の移動量

は小さく，地球での実用には非常に難しいように考えられる．しかし，地球よりも重力の小さな環境では，これらの問題は緩和されるため，実現可能な入力が得られると考えられる．

第4章　結言

本稿では，車両型ロボットを想定し，未知環境で動作する際の動作制御について提案した．未知の環境で動作する場合，ロボットには環境の認識や地図生成，自己位置推定といった基本的な技術だけでなく，環境に適応した動作制御が求められる．例えば，作業中に徐々に明らかになる環境や現在の状況に適した動作を選択する手法が考えられる．また，従来までロボットに要求されてきた動作よりも高度な動作を求める状況というのも考えられる．本稿では，2 つの環境に適応した動作制御手法を提案した．本稿で述べられている結果は，すでに文献 [1]-[3] に掲載されている．

2 章では，未知環境における掃引に関する動作制御手法を提案した．オンラインで生成される地図は，掃引初期から掃引終盤にかけて，徐々に信頼性を高めながら完成する．そのため，掃引終盤で効率が良い地図全体を利用した動作計画は，掃引初期には無駄になることが多い．また，ロボットは移動することで自己位置の確からしさが低下する．自己位置推定の結果と実際の位置が大きく違うと，掃引を行っても意味がない．そのときの状況に応じて，ロボットは動作制御の切り替えを行う必要がある．

そこで，新たな掃引動作制御手法，Gradually Building Map-based（GBM-based）アルゴリズムを提案した．GBM-based アルゴリズムは 3 つの動作制御手法と 2 つの条件分岐で構成される．3 つの動作制御はそれぞれ，地図周辺のみを利用する Local Reference-based，地図全体を利用する Global Reference-based，自己位置の確からしさを向上させる Landmark Search であり，「ロボット周辺に未掃引領域が存在するか」と「自己位置の確からしさが小さいか」という 2 つの条件によって各動作制御を選択する．2 つの条件分岐には，掃引の終了時刻や部屋環境の大きさに関する情報が入っていないにも関わらず，掃引開始から掃引終了にかけて徐々に Local Reference-based

から Global Reference-based へと選択する割合が切り替わる点が，本提案アルゴリズムの大きなポイントである．地図の完成度が高まるにつれ，ロボット周辺に未掃引領域が存在しない確率が大きくなるため，環境の大きさに依存することなく，掃引初期には Local Reference-based が主に，掃引終盤では Global Reference-based が主に選択されると考えられる．また，自己位置の確からしさは時間に依存しないとすると，Landmark Search は一定の割合で選択されると考えられる．

2 章では，GBM-based アルゴリズムを，掃除ロボットに適用した場合のアルゴリズムを示している．本アルゴリズムの有効性の確認として，地図を用いず螺旋動作，壁反射動作，壁追従動作を組み合わせた Behavior-based アルゴリズムと比較を行った．各アルゴリズムを用いて掃引するシミュレーションと実験を行い，シミュレーションと実験の両方の結果から，本提案アルゴリズムの掃引率が高いことを示した．

3 章では，動物界で様々な場面で利用されている尻尾を利用し，不整地環境において移動能力を向上させる動作制御手法を提案した．車両ロボットは，平地での移動効率が高いが，不整地環境では数多くの移動を妨げる事象が発生する．不整地の斜面や路面の凹凸においては，ロボットが復帰不可能な転倒状態に陥ってしまうことが挙げられる．また，地図上には存在しない経路の断裂や浸水により走行が不能となり，計画した経路を変更せざるを得ない状況も考えられる．

このような事態を打開するための機構として，本研究では，動物界で利用されている尻尾を挙げ，ロボットへの適用を考えた．本研究での提案は 3 つ存在する．1 つ目は，ロボット本体に大きな反力・反トルクを生み出すことが可能な瞬間的に形状を保存する尻尾．2 つ目は，尻尾を利用した車両型ロボットの跳躍モデル．3 つ目は，宙に浮いた目標位置姿勢に到達するための跳躍動作制御手法である．尻尾の剛性を切り替え，尻尾の形状を保つことで，尻尾の重心の運動を停止させることができる．尻尾の形状を瞬間的に保存することで，尻尾重心には運動方向とは逆向きの大きな慣性力が発生し，ロボット本体に反力・反トルクとなって伝達する．3 章では，剛体 1 リンクの尻尾をロボットに搭載した場合に比べて，大きな反力・反トルクをロボット本体に与えることができることを示した．本論文では，この瞬間的に形状を保つ機能を，形状保存（Shape Memorization）と呼んだ．形状保存による反力・反

トルクを利用し，従来 2 次元平面のみでしか軌道を計画できなかった車両ロボットを，3 次元の鉛直方向，つまりタイヤが地面から離れる動作を可能にする手法を提案した．跳躍動作を 2 つの状態（Stumbling Phase と Flight Phase）に分けて，それぞれのモデル化を行った．Stumbling Phase は前輪のみが接地した状態，Flight Phase では両輪が接地した状態を意味する．提案モデルでは，Stumbling Phase はロボットが前輪を中心とした振り子運動として，Flight Phase は 2 リンクの浮遊ロボットとして表現されている．さらに，これらのモデルを用いて，宙に浮いた目標位置姿勢に到達するための入力設計手法を提案した．本手法では，各 Phase での拘束条件と Phase 間の境界条件から，尻尾の形状保存のタイミング，ロボットの前輪が地面から離れるタイミング，浮遊時の尻尾根元アクチュエータの軌道の設計を行う．本手法が有効であるかを調べるため，数値シミュレーションにて目標位置姿勢に到達していることを確認した．

謝 辞

本研究の計画から論文の作成に至るまで御指導頂きました九州大学大学院工学研究院機械工学部門　山本元司教授に心より感謝いたします．また，本論文をまとめるにあたり，有益な御助言を賜りました九州大学大学院システム情報科学研究院情報知能工学部門　倉爪亮教授，九州大学大学院工学研究院機械工学部門　田原健二准教授に深く感謝いたします．本研究を進めるにあたり九州大学大学院工学研究院機械工学部門　菊植亮准教授，中島康貴助教，九州大学国際教育センター　Svinin Mikhail 教授には多大な御指導を頂きました．新屋幸喜技術職員には実験機の製作に御協力頂きました．そして，秘書の立山みどりさん，永富瑞穂さんには様々なサポートをして頂きました．ここに深く感謝いたします．

また，既に御卒業された方々も含めて，様々な面で御協力頂きました研究室の先輩方，同輩，後輩の皆さんへ感謝いたします．

ロボット教育を受け始めてから 15 年，これまでの人生をロボット製作に捧げ，多くを捨てて生きてきたと言っても過言ではありません．しかし，これだけの教育を受けながらも，大きな業績なく時が過ぎてしまったというのは誠に遺憾に感じております．今後は，自らが過去のロボット教育における成功例となれるよう，一層尽力したい所存です．最後に，自分の人生に大きく影響を及ぼして頂いた若干名の方々には，深く感謝いたします．

平成 28 年 3 月 岩本　憲泰

参考文献

[1] N. Iwamoto and M. Yamamoto, “Complete Coverage Motion Control Using Gradually Building Map”, *Proceedings of the 13th International Conference on Control, Automation, Robotics and Vision*, pp. 1917–1922, 2014.

[2] 岩本憲泰, 池田毅, 山本元司, “漸進的生成地図に基づく掃引ロボットのための効率的動作生成”, 日本機械学会論文集（*C*編）, vol. 81, no. 823, pp. 1–15, 2015.

[3] N. Iwamoto and M. Yamamoto, “Jumping Motion Control Planning for 4-Wheeled Robot with a Tail”, *Proceedings of the 2015 IEEE/SICE International Symposium on System Integration*, pp. 871–876, 2015.

[4] S. M. Lavelle, “Motion Planning: The Essentials”, *IEEE Robotics and Automation Magazine*, pp. 79–89, March, 2011.

[5] S. M. Lavelle, “Motion Planning: Wild Frontiers”, *IEEE Robotics and Automation Magazine*, pp. 79–89, June, 2011.

[6] J. O. Wallgrun, “Robot Mapping”, *Hierarchical Volonoi Graphs: Spatial Representation and Reasoning for Mobile Robots*, pp. 11–43, 2010.

[7] B. Scholkopf and H. Mallot, “View-Based Cognitive Mapping and Path Planning”, *Adaptive Behavior*, vol. 3, no. 3, pp. 311–348, 1995.

[8] M. O. Franz, B. Scholkopf, H. A. Mallot and H. H. Bulthoff, “Learning View Graphs for Robot Navigation”, *Autonomous Robots*, vol. 5, iss. 1, pp. 111–125.

[9] S. Werner, B. Krieg-Bruckner and T. Herrmann, “Modeling Navigational Knowledge by Route Graphs”, *Spatial Cognition II - Integrating Abstract Theories, Empirical Studies, Formal Methods, and Practical Apprications*, Springer, vol. 1849 of the series Lecture Notes in Computer Science, pp. 295–316, 2000.

[10] H. Choset, S. Walker, K. Eiamsa-Ard and J. Burdick, “Sensor-based Exploration: Incremental Construction of the Hierachical Generalized Voronoi Graph”, *International Journal of Robotics Research*, vol. 19, no. 2, pp. 126–148, 2000.

[11] E. Remolina and B. Kuipers, “Towards a General Theory of Topological Maps”, *Artificial Intelligence*, vol. 152, iss. 1, pp. 47–104, 2004.

[12] A. Elfes, “A Sonar-Based Mapping and Navigation System”, *IEEE Journal of Robotics and Automation*, vol. RA-3, no. 3, pp. 249–265, 1987.

[13] A. Elfes, “Using Occupancy Grids for Mobile Robot Perception and Navigation”, *Computer*, vol. 22, iss. 6, pp. 46–57, 1989.

[14] H. P. Moravec, “Sensor Fusion in Certainty Grids for Mobile Robots”, *AI Magagine*, vol. 9, no. 2, pp. 61–74, 1988.

[15] S. Thrun, “Robotic Mapping: A Survey”, *Exploring Artificial Intelligence in the New Millennium*, pp. 1–35, 2003.

[16] S. Thrun, W. Burgard and D. Fox, “Probabilistic Robotics”, MIT Press, 2005.

[17] R. E. Kalman, “A New Approach to Linear Filtering and Prediction Problems”, *Transactions of the ASME-Journal of Basic Engineering*, vol. 82, series D, pp. 35–45, 1960.

[18] 片山徹, “応用カルマンフィルタ”, 朝倉書店, 1983.

[19] J. S. Liu and R. Chen, “Sequential Monte Carlo Methods for Dynamic Systems”, *Journal of the American Statistical Association*, vol. 93, no. 443, pp. 1032–1044, 1998.

[20] M. K. Pitt and N. Shephard, “Filtering via Simulation: Auxiliary Particle Filters”, *Journal of the American Statistical Association*, vol. 94, no. 446, pp. 590–599, 1999.

[21] H. Durrant-Whyte and T. Bailey, “Simultaneous Localization and Mapping: Part I”, *IEEE Robotics and Automation Magazine*, pp. 99–108, June, 2006.

[22] T. Bailey and H. Durrant-Whyte, “Simultaneous Localization and Mapping (SLAM): Part II”, *IEEE Robotics and Automation Magazine*, pp. 108–117, September, 2006.

[23] iRobot Coporation, Vacuum Cleaning Robot, http://www.irobot.com/For-the-Home/Vacuum-Cleaning/Roomba.aspx, （参照日 2016 年 1 月 14 日）.

[24] 株式会社 東芝, ロボットクリーナー, http://www.toshiba.co.jp/living/lineup/cleaners/0163_k2_pic_01.html, （参照日 2016 年 1 月 14 日）.

[25] LG Electronics Inc., Vacuum Cleaner, http://www.lg.com/jp/vacuum-cleaner, （参照日 2016 年 1 月 14 日）.

[26] Neato Robotics, Inc., Robot Vacuums, https://neatorobotics.com/robot-vacuums/, （参照日 2016 年 1 月 14 日）.

[27] パナソニック株式会社, ロボット掃除機「ルーロ」MC-RS1, http://panasonic.jp/soji/rulo/, （参照日 2016 年 1 月 14 日）.

[28] Dyson, ロボット掃除機,
http://www.dyson.co.jp/dyson-vacuums/robot.aspx,
（参照日 2016 年 1 月 14 日）.

[29] シャープ株式会社, ロボット家電 COCOROBO,
http://www.sharp.co.jp/cocorobo/, （参照日 2016 年 1 月 14 日）.

[30] J. Palacín, T. Palleja, I. Valgañon, R. Pernia and J. Roca, "Measuring Coverage Performances of a Floor Cleaning Mobile Robot Using a Vision System", *Proceedings of the 2005 IEEE International Conference on Robotics and Automation*, pp. 4236–4241, 2005.

[31] T. Palleja, M. Tresanchez, M. Teixido and J. Palacin, "Modeling Floor-Cleaning Coverage Performances of Some Domestic Mobile Robots in a Reduced Scenario", *Robotics and Autonomous Systems*, vol. 58, pp. 37–45, 2010.

[32] 中村仁彦, 関口暁宣, "カオティック移動ロボット", 日本ロボット学会誌, vol. 15, no. 6, pp. 918–926, 1997.

[33] K. Fallahi and H. Leung, "A Cooperative Mobile Robot Task Assignment and Coverage Planning based on Chaos Synchronization", *International Journal of Bifurcation and Chaos*, vol. 20, no. 1, pp. 161–176, 2010.

[34] 荒木猛志, 山本元司, "掃引ロボットの確定的反射動作による掃引完全性に関する考察", 第 *31* 回日本ロボット学会学術講演会, 3H1-06, 2013.

[35] 荒木猛志, 山本元司, "掃引ロボットの確定的反射行動と掃引効率について -確率的行動との類似性とカオス性-", *2014* 年日本機械学会ロボティクスメカトロニクス部門学術講演会, 1P2-Q04, 2014.

[36] A. Zelinsky, R. A. Jarvis, J. C. Byrne and S. Yuta, "Planning Paths of Complete Coverage of an Unstructured Environment by a Mobile Robot", *Proceedings of International Conference on Advanced Robotics*, pp. 533–538, 1993.

[37] J. S. Oh, Y. H. Choi, J. B. Park and Y. F. Zheng, "Navigation of Cleaning Robots Using Triangular-Cell Map for Complete Coverage", *Proceedings of the 2003 IEEE International Conference on Robotics and Automation*, pp. 2006–2011, 2003.

[38] Y. Gabriely and E. Rimon, "Spanning-Tree Based Coverage of Continuous Areas by a Mobile Robot", *Annals of Mathematics and Artificial Intelligence*, vol. 31, pp. 77–98, 2001.

[39] D. Kurabayashi, J. Ota, T. Arai and E. Yoshida, "Cooperative Sweeping by Multiple Mobile Robots", *Proceedings of the 1996 IEEE International Conference on Robotics and Automation*, pp. 1774–1749, 1996.

[40] 深澤祐介, T. Chomchana, 太田順, 湯浅秀男, 新井民夫, 淺間一, "格子点配置を用いた自律移動ロボットによる環境掃引経路計画", 計測自動制御学会論文集, vol. 39, no. 11, pp. 1054–1060, 2003.

[41] E. U. Acar and H. Choset, "Robust Sensor-Based Coverage of Unstructured Environments", *Proceedings of the 2001 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 61–68, 2001.

[42] S. C. Wong and B. A. MacDonald, "A Topological Coverage Algorithm for Mobile Robots", *Proceedings of the 2003 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 1685–1690, 2003.

[43] C. Luo and S. X. Yang, "A Bioinspired Neural Network for Real-Time Concurrent Map Building and Complete Coverage Robot Navigation in Unknown Environments", *IEEE Transactions on Neural Networks*, vol. 19, no. 7, pp. 1279–1298, 2008.

[44] B. Yamauchi, "Frontier-Based Exploration Using Multiple Robots", *Proceedings of the Second International Conference on Autonomous Agents*, vol. 19, no. 7, pp. 1279–1298, 1998.

[45] T. Wattanavekin, T. Ogata, T. Hara and J. Ota, “Mobile Robot Exploration by Using Environmental Boundary Information”, *ISRN Robotics*, vol. 2013, article id 954619, pp. 1–11, 2013.

[46] C. Stachniss, “Robotic Mapping and Exploration”, *Springer Tracts in Advanced Robotics*, vol. 55, 2009. doi 10.1007/978-3-642-01097-2

[47] 上田武, 山本元司, “RFID を用いた床面掃引ロボットの掃引領域推定と掃引動作制御”, 第 *15* 回ロボティクスシンポジア予稿集, pp. 482-487, 2010.

[48] C. Stachniss, D. Hahnel and W. Burgard, “Exploration with Active Loop-Closing for FastSLAM”, *Proceedings of the 2004 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 1505–1510, 2004.

[49] 石川統, 黒岩常祥, 塩見正衞, 松本忠夫, 守隆夫, 八杉貞雄, 山本正幸, “生物学辞典”, 東京化学同人, 2010.

[50] M. M. Poter, D. Adriaens, R. L. Hatton, M. A. Meyers and J. MchKittrick, “Why the Seahorse Tail is Square”, *Science*, vol. 349, pp. 1–7, 2015.

[51] M. M. Poter, E. Novitskaya, A. B. Castro-Ceseña, M. A. Meyers and J. MchKittrick, “Highly Deformable Bones: Unusual Deformation Mechanisms of Seahorse armor”, *Acta Biomaterialia*, vol. 9, pp. 6763-6770, 2013.

[52] S. M. O’Connor, T. J. Dawson, R. Kram and J. M. Donelan, “The Kangaroo’s Tail Propels and Powers Pentapedal Locomotion”, *Biology letters*, pp. 1–4, doi 10.1098/rsbl.2014.0381, 2014.

[53] 犬塚則久, 恐竜ホネホネ学, 日本放送出版協会, 2006.

[54] V. M. Arbour, “Estimating Impact Forces of Tail Club Strikes by Ankylosaurid Dinosaurs”, *PLoS One*, vol. 4, iss. 8, doi 10.1371/journal.pone.0006738, 2009.

[55] F. Weishampel, 真鍋真, 藤原慎一, 松本涼子, 恐竜学入門 -かたち・生態・絶滅-, 東京化学同人, 2015.

[56] K. Takita, R. Hodoshima and S. Hirose, “Fundamental Mechanism of Dinosaur-like Robot TITRUS-II Utilizing Coupled Drive”, *Proceedings of the 2000 IEEE International Conference on Intelligent Robots and Systems*, pp. 1670–1675, 2000.

[57] K. Takita, T. Katayama and S. Hirose, “The Efficacy of the Neck and Tail of Miniature Dinosaur-like Robot TITRUS-III”, *Proceedings of the 2002 IEEE International Conference on Intelligent Robots and Systems*, pp. 2593–2598, 2002.

[58] K. Takita, T. Katayama and S. Hirose, “Development of Dinosaur-like Robot TITRUS -its Dynamics and the Motion Utilizing the Dynamic Effect of the Neck and Tail-”, *Proceedings of the 2003 IEEE International Conference on Intelligent Robots and Systems*, pp. 607–612, 2003.

[59] 藤田雅博, “ペット型ロボットの感性表現”, 日本ロボット学会誌, vol. 17, no. 7, pp. 947–951, 1999.

[60] I. Leite, C. Martinho and A. Paiva, “Social Robots for Long-Term Interaction: A Survey”, *International Journal of Social Robotics*, vol. 5, iss. 2, pp. 291–308, 2013.

[61] A. Jusufi, D. I. Goldman, S. Revzen and R. J. Full, “Active Tails Enhance Arboreal Acrobatics in Geckos”, *PNAS*, vol. 105, no. 11, 2008.

[62] T. R. Kane and M. P. Scher, “A Dynamical Explanation of the Falling Cat Phenomenon”, *International Journal of Solid Structures*, vol. 5, pp. 663–670, 1960.

[63] 河村隆, 山藤和男, 小林剛, “空中を落下するロボットの姿勢制御と軟着地 （第1報, ねこひねりによる空中姿勢制御）”, 日本機械学会論文集（*C*編）, vol. 57, no. 544, pp. 3895–3900, 1991.

[64] 中村仁彦, “非ホロノミックロボットシステム 第4回 動力学的な非ホロノミック拘束の下での運動”, 日本ロボット学会誌, vol. 11, no. 7, pp. 999-1005, 1993.

[65] 河村隆, “ねこひねりロボット”, 計測と制御, vol. 36, no. 6, pp.413–414, 1997.

[66] A. Arabyan and D. Tsai, “A Distributed Control Model for the Air-Righting Reflex of a Cat”, *Biological Cybernetics*, vol. 79, iss. 5, pp. 393-401, 1998.

[67] 掃部雅幸, 吉田和夫, “ひねり宙返り運動の力学解析”, 日本機械学会論文集（*C*編）, vol. 69, no. 680, pp.1072–1079, 2003.

[68] 甲斐健也, 環一穂, “初期角運動量をもつ2剛体宇宙ロボットに対する3次元姿勢準最適制御”, 計測自動制御学会論文集, vol. 45, no. 6, pp. 320–326, 2009.

[69] D. A. McDonald, “How Does a Cat Fall on Its Feet”, *New Scientist*, vol. 7, no. 189, pp. 1647–1649, 1960.

[70] A. Jusufi, D. T. Kawano, T. Libby and R. J. Full, “Righting and Turning in Mid-Air Using Appendage Inertial: Reptile Tails, Analytical Models and Bio-Inspired Robots”, *Bioinspiration and Biomimetics*, vol. 5, no. 4, pp. 1–12, 2010.

[71] E. Chang-Siu, T. Libby, M. Tomizuka and R. J. Full, “A Lizard-Inspired Active Tail Enables Rapid Maneuvers and Dynamic Stabilization in a Terrestrial Robot”, *Proceedings of the 2011 IEEE International Conference on Intelligent Robots and Systems*, pp. 1887–1894, 2011.

[72] T. Libby, T. Y. Moore, E. Chang-Siu, D. Li, D. J. Cohen, A. Jusufi and R. J. Full, “Tail-Assisted Pitch Control in Lizards, Robots and Dinosaurs”, *Nature letter*, vol. 481, pp. 181-184, 2012.

[73] A. M. Johnson, T. Libby, E. Chang-Siu, M. Tomizuka, R. J. Full and D. E. Koditschek, "Tail Assisted Dynamic Self Righting", *Proceedings of the 15th International Conference on Climbing and Walking Robots*, pp. 611–620, 2012.

[74] E. Chang-Siu and T. Libbym, M. Brown, R. J. Full and M. Tomizuka, "A Nonlinear Feedback Controller for Aerial Self-Righting by a Tailed Robot", *Proceedings of the 2013 IEEE/RSJ International Conference on Robotics and Automation*, pp. 32–39, 2013.

[75] J. Zhao, T. Zhao, N. Xi, F. J. Cintron, M. W. Mutka and L. Xiao, "Controlling Aerial Maneuvering of a Miniature Jumping Robot Using Its Tail", *Proceedings of the 2013 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 3802–3807, 2013.

[76] A. De and D. E. Koditschek, "Parallel Composition of Templates for Tail-Energized Planar Hopping", *Proceedings of the 2015 IEEE International Conference on Robotics and Automation*, pp. 4562–4569, 2015.

[77] Pullin, Andrew O and Kohut, Nicholas J and Zarrouk, David and Fearing, Ronald S, "Dynamic Turning of 13 cm Robot Comparing Tail and Differential Drive", *Proceedings of the 2012 IEEE International Conference on Robotics and Automation*, pp. 5086–5093, 2012.

[78] C. Fisher and A. Patel, "FlipBot: A Lizard Inspired Stunt Robot", *Preprints of the 19th World Congress the International Federation of Automatic Control*, pp. 4837–4842, 2014.

[79] R. Briggs, J. Lee, M. Hanberland and S. Kim, "Tails in Biometic Design: Analysis, Simulation and Experiment", *Proceedings of the 2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 1473–1480, 2012.

[80] A. Patel and M. Braae, “Rapid Turning at High-Speed: Inspirations from the Cheetah’s Tail”, *Proceedings of the 2013 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 5506–5511, 2013.

[81] A. Crespi, A. Badertscher, A. Guignard and A. J. Ijspeert, “Swimming and Crawling with an Amphibious Snake Robot”, *Proceedings of the 2005 IEEE International Conference on Robotics and Automation*, pp. 3024–3028, 2005.

[82] A. Crespi and A. J. Ijspeert, “Salamandra Robotica: a Biologically Inspired Amphibious Robot that Swims and Walks”, *Artificial Life Models in Hardware*, pp. 35–64, 2009.

[83] K. A. McIsaac and J. P. Ostrowski, “Experiments in Closed-Loop Control for an Underwater Eel-like Robot”, *Proceedings of the 2002 IEEE International Conference on Robotics and Automation*, vol. 1, pp. 750–755, 2002.

[84] E. Kim and Y. Youm, “Design and Dynamic Analysis of Fish Robot: Potuna”, *Proceedings of the 2004 IEEE International Conference on Robotics and Automation*, vol. 5, pp. 4887–4892, 2004.

[85] J. Liu and H. Hu, “Biological Inspiration: From Carangiform Fish to Multi-Joint Robotic Fish”, *Journal of Bionic Engineering*, vol. 7, no. 1, pp. 35–48, 2010.

[86] A. J. Clark, J. M. Moore, J. Wang, X. Tan and P. K. McKinley, “Evolutionary Design and Experimental Validation of a Flexible Caudal Fin for Robotic Fish”, *Artificial Life*, vol. 13, pp. 325–332, 2012.

[87] H. Choset, “Coverage for Robotics - A Survey of Recent Results”, *Annals of Mathematics and Artificial Intelligence*, vol. 31, pp. 113–126, 2001.

[88] H. Choset and P. Pignon, “Coverage Path Planning: The Boustrophedon Decomposition”, *Proceedings of International Conference on Field and Service Robotics*, 1997.

[89] 布施嘉裕, 丹沢勉, 清弘智昭, “全方向移動床磨きロボットへの非干渉 PID 制御の適用”, 日本ロボット学会誌, vol. 27, no. 6, pp. 679–684, 2009.

[90] E. González, O. Álvarez, Y. Díaz, C. Parra and C. Bustacara, “BSA: A Complete Coverage Algorithm”, *Proceedings of the 2005 IEEE International Conference on Robotics and Automation*, pp. 2040–2044, 2005.

[91] H.-H. Bock, “Clustering Methods : A History of k-Means Algorithms”, *Selected Contributions in Data Analysis and Classification*, Part of the series Studies inClassification, Data Analysis, and Knowledge Organization, pp. 161–172, 2007.

[92] 堀田大輔, 川田浩彦, 大矢晃久, 油田信一, “測域センサを応用した複数ランドマークの認識による大域的自己位置推定”, *Proceedings of the JSME Conference on Robotics and Mechatronics*, 2P2-C16, 2008.

[93] K. Kodaka, H. Niwa, Y. Sakamoto, M. Otake, Y. Kanemori and S. Sugano, “Pose Estimation of a Mobile Robot on a Lattice of RFID Tags”, *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 1385–1390, 2008.

[94] 倉林大輔, 新井民夫, 岩瀬寛司, 太田順, “地図誤差に動的に適応する移動ロボットの掃引作業計画”, 日本ロボット学会誌, vol. 17, no. 5, pp. 677–684, 1999.

[95] 太田順, 倉林大輔, 新井民夫, “知能ロボット入門”, コロナ社, pp. 119–120, 2001.

[96] P. ブレモー, 釜江哲朗, 向井久, “モデルで学ぶ確率入門”, Springer, 1992.

[97] 美多勉, “非線形制御入門　-劣駆動ロボットの技能制御論-”, 昭晃堂, 2000.

[98] M. D. Berkemeier and R. S. Fearing, "Sliding and Hopping Gaits for the Underactuated Acrobot", *IEEE Transactions on Robotics and Automation*, vol. 14, no. 4, pp. 629–634, 1998.

[99] T. S. Majmudar, M. Sperl, S. Luding and R. P. Behringer, "Jamming Transition in Granular Systems", *Physical Review Letters*, vol. 98, iss. 5, pp.1–4, 2007.

[100] Y.-J. Kim, S. Cheng, S. Kim and K. Iagnemma, "A Novel Layer Jamming Mechanism With Tunable Stiffness Capability for Minimally Invasive Surgery", *IEEE Transactions on Robotics*, vol. PP, no. 99, pp. 1–12, 2013.

[101] Xin, Xin and Mita, Tsutomu and Kaneda, Masahiro, "The Posture Control of a 2-Link Free Flying Acrobot with Initial Angular Momentum", *Proceedings of the 41th International Conference on Decision and Control*, vol. 2, pp. 2068–2073, 2002.

[102] 服部 邦雄，山浦 弘，小野 京右，"浮遊ロボットの姿勢制御（トルク入力の導出と時変ゲインによる最大トルクの低減）"，日本機械学会論文集（*C*編），vol. 70，no. 691, pp. 782–789, 2004.

[103] K. Hattori, H. Yamaura and K. Ono, "Torque-based aerial posture control of a two-link acrobat robot", *Proceedings of the Institution of Mechanical Engineers. Part I: Journal of Systems and Control Engineering*, vol. 218, no. 7, pp. 595–601, 2004.

[104] X. Xiong, R. Kikuuwe and M. Yamamoto, "A Differential Algebraic Method to Approximate Nonsmooth Mechanical Systems by Ordinary Differential Equations", *Journal of Applied Mathematics*, vol. 2013, article 320276, 2013.