解 説

多分解能状態空間ロードマップ法による軌道計画

Trajectory Planning Using Multi-resolution State Roadmaps

田 崎 勇 一* *名古屋大学

Yuichi Tazaki* *Nagoya University

1. は じ め に

ロボットの軌道生成に関する研究が過去半世紀に渡り精力的に取り組まれてきた。そのなかに、ロボットが移動可能な領域とその接続関係をグラフとして表現するロードマップ法がある。ロードマップとはロボット自身が作業空間上を自由に移動するための「地図」であり、一度作成しておけば任意の二点間を結ぶ軌道を短時間で生成することが可能となる。

ロードマップの生成法として、古くは可視グラフや一般化ボロノイグラフ、シルエット法など作業環境の幾何学的特徴をもとにグラフ構造を抽出する方法が提案された(文献[1][2])が、実用的に適用可能な空間は二次元あるいは三次元に限られていた。より高次元のコンフィギュレーション空間への適用を指向し、Probabilistic Roadmap (PRM) 法が提案された[3]。PRM 法は、空間中のランダムな位置にノードを生成しつつ、局所計画器(local planner)を用いて移動可能と判定された二点間をリンクで接続することでロードマップを生成する。PRM 法は実装や拡張が容易であることから、ノードのサンプル方法や局所計画器、リンクの接続規範などに関して様々なバリエーションが存在し、その応用範囲も広い。また、PRM と呼ばれる手法の中にも multi-query PRM と single-query PRM が存在する。前者は一つのロードマップで複数の経路生成要求(query)に対応可能なものを生成するが、後者はロボットの現在位置を始点として決められた目標点への経路を生成する過程で PRM を作成する。このため、後者は「地図」としての性質を持たないという意味において狭義のロードマップ法には含まれないと言える。

多くのロボットは移動可能な方向が微分方程式などによって制約されている。障害物回避に加えてこのような移動制約を考慮した軌道生成問題は kinodynamic planning と呼ば

れている。移動制約を考慮した手法としては前述の single-query PRM に基づく手法[6]や、Rapidly exploring random tree (RRT) など(文献[4][5])がある。これらの手法は、いずれもロボットの現在位置を根とする木構造を成長させることで目標点までの連結な経路を探索する手法である。文献[6]では既存の木ノードと制御入力をランダムに選択して枝を伸ばすのに対して、文献[5]ではまず目標点をランダムに生成し、その最近傍ノードから目標点へ向かって枝を伸ばすところに違いがある。これらのオンライン手法は、近年の計算リソースの飛躍の向上に合わせて広く用いられるようになってきた。しかしながら、移動制約を考慮しつつ複数の経路生成に利用可能(multi-query)なロードマップの生成を議論した文献は少ない。

筆者らは、移動制約を考慮したロードマップの生成法として多分解能状態ロードマップ法(Multi-resolution state roadmap method)を提案し、軌道生成問題[10]やマルチロボットのスケジューリング問題[11]、自動駐車[12]へ応用してきた。状態ロードマップの各ノードは作業空間上の位置を表す点と、それ以外の状態(速度など)の区間を保持する。また、ロードマップ中の二つのノードは、それらが保持する区間に含まれる任意の状態の組み合わせについて制約を満たす軌道が得られるとき有向リンクで接続される。所定の終了条件が満たされるまで各ノードが保持する状態の区間を細分化することにより、状態空間上に複数の解像度レベルを持つロードマップが生成される。多分解能(multi-resolution)を銘打った環境表現や経路生成法はいくつか提案されている(文献[7]~[9])が、その多くは障害物形状の複雑さや障害物への距離などに基づいて表現粒度を変化させるものであり、ロボットの移動制約を多分解能表現に結び付けている点で本手法はこれらと異なる。本稿では、状態ロードマップ法の原理を簡潔に述べ、自動走行車の駐車軌道生成に適用した例を紹介する。

2. 状態ロードマップ

本章では状態ロードマップ法の概要を説明する。通常、状態ロードマップはオフラインで作成しておき、ロボットに記憶しておくことでオンラインでの軌道生成に利用する。

原稿受付 2014 年 5 月 19 日

キーワード: Path Planning, Roadmap Method, Multi-resolution

*〒 464-8603 名古屋市中種区不老町

*Chikusa-ku, Nagoya-shi, Aichi

2.1 二点境界値問題の解析解

はじめに、以下のような一般的な常微分方程式で表現されるロボットのダイナミクスを考える。

$$\dot{x} = f(x, u). \quad (1)$$

ここで、状態は $x = [p^\top \theta^\top]^\top$ であり、 $p \in \mathcal{P}$ は作業空間上の位置を表す（本稿では $\mathcal{P} \subset \mathbb{R}^2$ とする）。また、 $\theta \in \Theta$ は位置以外の状態量を表し、これを拡張状態と呼ぶ。

状態ロードマップ法を適用するための要求条件として、上の常微分方程式の二点境界値問題の解析解が与えられている必要がある。つまり、与えられた二状態 x, x' および付加的なパラメータ $\alpha \in \mathcal{A}$ に対し、移動時間 T 、制御入力信号 $u(t)$ と状態の軌道 $x(t)$ として $x(0) = x, x(T) = x'$ および $t \in [0, T]$ について式 (1) を満たすものが得られなければならない。これらが x, x' および α によってパラメータ表現されていることを強調するため $T(x, x', \alpha)$ 、 $u(t|x, x', \alpha)$ および $x(t|x, x', \alpha)$ と書く。付加パラメータ α の物理的な意味は問題ごとに異なる。また、付加パラメータを持たない場合や、離散値をとる場合も考えられる。

加えて、上述の軌道は任意の時刻で諸々の制約条件を満たす必要がある。具体的な制約条件は問題ごとに異なるが、ここでは一般的に制約条件を満たす状態と制御入力の集合を安全領域 $\mathcal{S}$ と表すと、満たすべき条件は下記となる。

$$(x(t|x, x', \alpha), u(t|x, x', \alpha)) \in \mathcal{S} \quad \forall t \in [0, T(x, x', \alpha)] \quad (2)$$

二状態と付加パラメータの組 (x, x', α) によって与えられる軌道が式 (2) を満たすとき、 (x, x', α) は実行可能であるという。また、位置の対 (p, p') に対し、 $((p, \theta), (p', \theta'), \alpha)$ が実行可能であるような $(\theta, \theta', \alpha)$ の集合を実行可能集合と呼び、 $\mathcal{F}(p, p')$ と表す。

2.2 状態ロードマップの構築

状態ロードマップは、位置空間上に定義されたロードマップを元にして構築する。位置ロードマップは無向グラフ $\mathcal{M}^p = (\mathcal{N}^p, \mathcal{L}^p)$ で表される。ここで $\mathcal{N}^p \subset \mathcal{P}$ は位置ノードの集合、 $\mathcal{L}^p \subseteq \mathcal{N}^p \times \mathcal{N}^p$ は無向リンクの集合である。状態ロードマップは有向グラフ（厳密には状態遷移系と呼ぶのが正しい） $\mathcal{M}^x = (\mathcal{N}^x, \mathcal{L}^x, \rightarrow)$ で表される。ここで $\mathcal{N}^x$ は状態ノードの集合、 $\mathcal{L}^x$ は遷移ラベルの集合である。状態ノードは $n = (p, [\theta])$ で表され、 $p \in \mathcal{N}^p$ は位置を表す点、 $[\theta] \subset \Theta$ は拡張状態の区間である。あるノード n について、それに付随する位置と拡張状態区間をそれぞれ p_n と $[\theta_n]$ と書く。遷移ラベルは $l = ((p, p'), [\alpha])$ で表され、 (p, p') は対応する位置の対、 $[\alpha]$ は付加パラメータの区間を表す。与えられた l に対して、それぞれを $(p, p')_l$ 、 $[\alpha_l]$ と書く。また、 $\rightarrow \subset \mathcal{N}^x \times \mathcal{N}^x \times \mathcal{L}^x$ は遷移関係を表し、

図1 状態ロードマップの模式図

$(n, n', l) \in \rightarrow$ のとき $n \xrightarrow{l} n'$ と書く。さらに遷移には縦断遷移と横断遷移の二種類がある。縦断遷移は、 $p_n = p_{n'}$ であり $[\theta_n]$ と $[\theta_{n'}]$ が隣接する解像度レベルにあるような (n, n') に定義される。縦断遷移は純粹に解像度の切り替えを表し、物理的な移動を表すものではない。他方、横断遷移は $(p_n, p_{n'}) = (p, p')_l$ で $[\theta_n] \times [\theta_{n'}] \times [\alpha_l] \subseteq \mathcal{F}(p_n, p_{n'})$ を満たす (n, n', l) に定義される。図1は状態ロードマップを模式的に示したものである。ただし、図では簡単のために付加パラメータ α がない場合を示している。各位置 $p \in \mathcal{N}^p$ について、拡張状態の区間の集まりが二分木状に保持されており、 p と一つ一つの区間との対が状態ロードマップのノードを表す。また、隣接する解像度レベル間の縦方向の矢印が縦断遷移を表し、同一レベル間の横方向の矢印が横断遷移を表す。

以下に説明するアルゴリズムによって、与えられた位置ロードマップを元にして状態ロードマップを構築することができる。一般に、領域分割の適切な解像度は事前に分からない。一方では、様々な始点と終点を実行可能な軌道で結ぶために十分高い解像度を設定する必要がある。他方では、解像度は状態ロードマップを用いた経路計画の計算コストに大きく影響する。したがって、必要最低限度の解像度を見つけることが重要となる。この問題に対し、段階的に解像度を高めながら繰り返し領域を分割していく方法を採用する。また、状態ロードマップの元となる位置ロードマップの設計も同様に重要であるが、これは設計者が定め、アルゴリズムに与えることとする。

状態ロードマップの構築手順を Algorithm 1 に示す。アルゴリズムへは位置ロードマップ $\mathcal{M}^p = (\mathcal{N}^p, \mathcal{L}^p)$ および最大解像度 $N^\top$ が入力される。はじめに、解像度レベル1の状態ノード集合および遷移ラベル集合を以下のように定義する。

$$\begin{aligned} \mathcal{N}_1^x &= \bigcup_{p \in \mathcal{N}^p} (p, \Theta), \\ \mathcal{L}_1^x &= \bigcup_{(p, p') \in \mathcal{L}^p} ((p, p'), \mathcal{A}) \end{aligned} \quad (3)$$

なんらかの事前知識に基づき、 Θ および $\mathcal{A}$ がある程度分

Algorithm 1 状態ロードマップ構築アルゴリズム

```

1: Input  $\mathcal{M}^p, N^r$ 
2: Define  $\mathcal{N}_1^x, \mathcal{L}_1^x$  using (3),  $\rightarrow := \emptyset$ 
3: for  $i \in [1 : N^r - 1]$  do
4:   // 実行可能性の判定
5:    $\mathcal{N}^{\text{div}} := \emptyset$ 
6:    $\mathcal{L}^{\text{div}} := \emptyset$ 
7:   for  $(n, n', l) \in \mathcal{N}_i^x \times \mathcal{N}_i^x \times \mathcal{L}_i^x, (p_n, p_{n'}) = (p, p')_l$  do
8:     if  $[\theta_n] \times [\theta_{n'}] \times [\alpha_l] \subseteq \mathcal{F}(p_n, p_{n'})$  then
9:       // 横断遷移の定義
10:       $\rightarrow_i := \rightarrow_i \cup (n, n', l)$ 
11:     else if  $[\theta_n] \times [\theta_{n'}] \times [\alpha_l] \cap \mathcal{F}(p_n, p_{n'}) \neq \emptyset$  then
12:        $\mathcal{N}^{\text{div}} := \mathcal{N}^{\text{div}} \cup \{n, n'\}$ 
13:        $\mathcal{L}^{\text{div}} := \mathcal{L}^{\text{div}} \cup \{l\}$ 
14:     end if
15:   end for
16:   // 区間の分割
17:    $\mathcal{N}_{i+1}^x := \emptyset$ 
18:   for  $n \in \mathcal{N}^{\text{div}}$  do
19:      $n_1 := (p_n, [\theta_n, \frac{\theta_n + \bar{\theta}_n}{2}])$ 
20:      $n_2 := (p_n, [\frac{\theta_n + \bar{\theta}_n}{2}, \bar{\theta}_n])$ 
21:      $\mathcal{N}_{i+1}^x := \mathcal{N}_{i+1}^x \cup \{n_1, n_2\}$ 
22:     // 縦断遷移の定義
23:      $\downarrow_{i+1}^i := \downarrow_{i+1}^i \cup \{(n, n_1), (n, n_2)\}$ 
24:      $\uparrow_{i+1}^i := \uparrow_{i+1}^i \cup \{(n_1, n), (n_2, n)\}$ 
25:   end for
26:    $\mathcal{L}_{i+1}^x := \emptyset$ 
27:   for  $l \in \mathcal{L}^{\text{div}}$  do
28:      $l_1 := ((p, p')_l, [\underline{\alpha}_l, \frac{\underline{\alpha}_l + \bar{\alpha}_l}{2}])$ 
29:      $l_2 := ((p, p')_l, [\frac{\underline{\alpha}_l + \bar{\alpha}_l}{2}, \bar{\alpha}_l])$ 
30:      $\mathcal{L}_{i+1}^x := \mathcal{L}_{i+1}^x \cup \{l_1, l_2\}$ 
31:   end for
32:    $\mathcal{N}^x := \bigcup_i \mathcal{N}_i^x, \mathcal{L}^x := \bigcup_i \mathcal{L}_i^x,$ 
33:    $\rightarrow := \bigcup_i \rightarrow_i \cup \bigcup_i \uparrow_{i+1}^i \cup \bigcup_i \downarrow_{i+1}^i$ 
34: end for

```

割された状態からはじめても構わない。これをもとに、解像度レベル i を 1 から $N^r - 1$ まで変化させながら各解像度レベルの状態ノード、遷移ラベルおよび遷移関係を生成していく。各解像度レベル i について、まず横断遷移が定義可能か、あるいは領域をさらに分割する必要があるかを判定する。この際、 $[\theta_n] \times [\theta_{n'}] \times [\alpha_l]$ と $\mathcal{F}(p_n, p_{n'})$ の関係性に基いて判定を行う。もし $[\theta_n] \times [\theta_{n'}] \times [\alpha_l]$ が $\mathcal{F}(p_n, p_{n'})$ に含まれている場合、横断遷移 $n \xrightarrow{l} n'$ を定義する。この場合は解像度レベル i で実行可能性の判定ができたため、これ以上分割を施す必要はない。他方、 $[\theta_n] \times [\theta_{n'}] \times [\alpha_l]$ が $\mathcal{F}(p_n, p_{n'})$ と交差するが完全には含まれていない場合が考えられる。このとき、領域内の中に実行可能な軌道を与える状態対と実行不可能な軌道を与えるものとが混在していることになるため、領域分割の解像度を引き上げる必要がある。したがって、 n, n' および l は分割対象となる状態ノードおよび遷移ラベルを格納する集合 $\mathcal{N}^{\text{div}}$ および $\mathcal{L}^{\text{div}}$ にそれぞれ格納される。最後に、もし $[\theta_n] \times [\theta_{n'}] \times [\alpha_l]$ が $\mathcal{F}(p_n, p_{n'})$ と交差しない場合、領域内のいかなる状態対も実行可能な軌道を与えないことになる。このため、横断遷移を定義せず、また区間をこれ以上分割する必要もない。各反復ステップの最後に、集合

$\mathcal{N}^{\text{div}}$ および $\mathcal{L}^{\text{div}}$ に格納された要素に対応する区間が分割される。まず各 $n \in \mathcal{N}^{\text{div}}$ に対し、 $[\theta_n]$ が二つの部分領域に等分割される。次に、二つの状態ノード n_1 と n_2 が生成され、これらに位置 p_n と分割された部分領域のそれぞれが割り当てられる。また、 n と n_1, n_2 の間には縦断遷移が双方向に定義される。同様に、各 $l \in \mathcal{L}^{\text{div}}$ に対し、 $[\alpha_l]$ を等分割し、あらたに遷移ラベル l_1 と l_2 が生成される。

上で述べたアルゴリズムの重要な要素の一つに、実行可能性判定がある。領域 $[\theta] \times [\theta'] \times [\alpha]$ は無限集合であり、また $\mathcal{F}$ は一般には非線形関数であるため、交差と包含性を厳密に判定することは難しい。そこで、 $[\theta], [\theta']$ および $[\alpha]$ は区間であり、したがって $[\theta] \times [\theta'] \times [\alpha]$ も区間であることに注目する。これに加えて、もし $\mathcal{F}(p, p')$ が凸集合であるならば、 $[\theta] \times [\theta'] \times [\alpha]$ と $\mathcal{F}(p, p')$ の交差と包含性の判定は $[\theta] \times [\theta'] \times [\alpha]$ の各端点が $\mathcal{F}(p, p')$ に含まれるかと調べるだけでよい。このようにすれば、領域の実行可能性判定は有限個の点の実行可能性判定に帰着する。残念ながら、 $\mathcal{F}(p, p')$ は多くの具体的な問題において非凸であることが分かっている。したがって、端点のみを調べる方法では領域全体に対する実行可能（不能）性の保証にはならない。一つの対処方法としては、 $\mathcal{F}(p, p')$ の下近似となるような凸集合を考え、これを判定に用いることが考えられる。この方法は多少保守的であるが、実行不能な点を可能と誤判定してしまう恐れはないという意味では安全な方法である。また、下近似を構成する方法は個別の問題に依存するため、本稿では述べない。

2.3 状態ロードマップを用いた軌道生成

始点 (p^s, θ^s) と終点 (p^g, θ^g) を結ぶ軌道を生成する方法について述べる。このとき、始点および終点が状態ロードマップのいずれかのノードに含まれるとは限らないため、工夫が必要となる。まず、一時的なノード $n^s = (p^s, \theta^s)$ を生成し、これを状態ロードマップに加える。次に、 n^s とその他の各ノード $n \in \mathcal{N}^x$ について実行可能性 $\theta^s \times [\theta_n] \in \mathcal{F}(p^s, p_n)$ を判定し、もしこれに合格したら n^s から n への横断遷移を生成する。終点に対しても $n^g = (p^g, \theta^g)$ を加え、同様の操作を行う。

次のステップとして、状態ロードマップ上で n^s から n^g への最短経路を計画する。ここで、各遷移のコストは以下のように定義する。まず、縦断遷移は物理的な移動を伴わないため、そのコストは 0 とする。また、横断遷移 (n, n', l) については、 $(p_n, m([\theta_n])), (p_{n'}, m([\theta_{n'}])), m[\alpha_l]$ で定義される軌道の物理的コストを評価する。ただし $m([\theta])$ は区間 $[\theta]$ の中点である。第 3.1 節の例のように付加パラメータ α が移動時間を表す場合は、 $m[\alpha]$ をコストとすることで最短時間軌道計画となる。最短経路の計算にはダイクストラ法などの古典的な探索アルゴリズムを適用できるが、上向きと下向きの縦断遷移が連続すると実行可能な軌

道が対応しない場合があるため、これを除くために工夫が必要である。

3. 適 用 例

3.1 慣性を持つ全方向ロボットの軌道生成

加速度を入力とする全方向移動型ロボットの軌道計画への応用例を示す。環境の模式図を図2に示す。二次元平面上の有界な領域が障害物と自由領域に分かれており、自由領域内にいくつかの経路点が配置されている。二つの経路点を結ぶ直線分が自由領域の内部に含まれるとき、経路点同士は無向リンクで接続される。ロボットの目的は、任意に指定された始点から終点へ向かって障害物に衝突せずに移動することである。ロボットのダイナミクスは

$$\ddot{p}(t) = u(t) \quad (4)$$

であり、 $p(t)$ と $u(t)$ はそれぞれロボットの位置と加速度入力である。また、ロボットの速度を $v(t) = \dot{p}(t)$ として状態を $x(t) = [p(t) \ v(t)]^T$ とする。与えられた二状態 $x = [p \ v]^T$ と $x' = [p' \ v']^T$ を時間 T で移動する二点境界値問題を考える。加速度入力の時間関数としては、

$$u(t) = \begin{cases} u_1 & \text{if } t \in [0, T/2], \\ u_2 & \text{otherwise} \end{cases} \quad (5)$$

で与えられる二段階のステップ関数を考える。すると、未知数 u_1 および u_2 は境界条件より

$$\begin{aligned} u_1 &= \frac{4}{T^2}(p' - p) - \frac{1}{T}(v' + 3v), \\ u_2 &= -\frac{4}{T^2}(p' - p) + \frac{1}{T}(3v' + v) \end{aligned}$$

と得られ、その結果として軌道が一意に定まる。軌道に課す拘束条件としては、速度制約、加速度制約および障害物回避を考える。まず速度・加速度制約は

$$\begin{cases} |v(t)| \leq V^{\max}, \\ |u(t)| \leq U^{\max} \end{cases} \quad \forall t \in [0, T] \quad (6)$$

図2 環境のモデル

となる。障害物回避制約は、単純に移動する二点を結ぶ直線分からの逸脱量に関する制約として下記のように記述する。

$$\begin{cases} \tau^T(p(t) - p) \geq -\epsilon, \\ \tau^T(p(t) - p') \leq \epsilon, \\ \eta^T(p(t) - p) \in [-w, w] \end{cases} \quad \forall t \in [0, T] \quad (7)$$

ここで τ は $p' - p$ と同じ向きの単位ベクトル、 η は τ に直交する単位ベクトルである。また、 ϵ は線分に沿った方向の許容逸脱量、 w は線分に直交する方向の許容逸脱量（道幅）を表す定数である。

上述の問題について状態ロードマップ法を適用する場合、速度 v と移動時間 T がそれぞれ拡張状態と移動パラメータとなる。また、制約条件 (6) (7) は (v, v', T) について凸性を持たないため工夫が必要である（詳しくは文献[10]を参照）。

図2の環境に状態ロードマップを構築し、点Aから点Gへ移動する軌道を計画した結果を図3に示す。各点を中心とする二次元の矩形は速度 v の分割を表している。また、各点間にある縦方向の一次元の区間は、移動時間 T の分割を表している。計画された軌道は制約 (6) (7) を満たしており、また状態ロードマップの解像度のレベルで近似的に最短時間の軌道となっている。

3.2 ノンホロノミック特性を移動ロボットの軌道生成と自動駐車への応用

本節ではノンホロノミック特性を持つ車両型ロボットの軌道計画と、その自動駐車への応用例を示す。ロボットの状態方程式は次のようになる。

$$\begin{aligned} \dot{x}(s) &= \cos(\theta(s)), & \dot{y}(s) &= \sin(\theta(s)), \\ \dot{\theta}(s) &= c(s), & \dot{c}(s) &= u(s). \end{aligned} \quad (8)$$

ここで、 s は軌道に沿った符号付きの距離を表し、始点において0をとり、前進すると増加、後退すると減少する。また、 $p(s) = [x(s), y(s)]^T$, $\theta(s)$ および $c(s)$ は s における位置、向きおよび曲率を表す。さらに、 $u(s)$ は制御入力で、 s に関する曲率の変化率を表す。

このようなノンホロノミック特性を持つロボットの二点

図3 状態ロードマップと点A(0,0)から点G(1500,0)への軌道

(a) F 軌道

(b) FF 軌道

図 4 二種類の前進軌道

図 5 適用可能な軌道の種類に基づく θ - θ' 空間の分割

境界値問題の解はクロソイド曲線と線分の組み合わせで表現できる. 具体的には, 状態の組 (p, θ) と (p', θ') を結ぶ軌道は, 前進・後退, および始点と終点との位置・向きに関係に基づき四種類が考えられる. 図 4 に, そのうちの二種類 (F, FF) を示す. 残りの二種類 (R, RR) も前進・後退の違いのみで形状は同じである. 詳細は文献 [12] を参照されたい. これら四種類のいずれが適用可能かは θ と θ' の関係性で決まる. 図 5 に, 適用可能な軌道の種類に基づき θ - θ' 平面を分割した図を示す.

軌道に課す拘束条件としては, 曲率制約と障害物回避を考える. 曲率は軌道に沿って区分的線形な変化を示すので, クロソイド片あるいは線分の端点において極値をとる. したがって, これらの極値において最大曲率を超過しないかを判定すればよい. 一方, 衝突回避制約については, 前節のように単純に道幅に関する範囲制約では駐車のような大きな曲率を持つ軌道の計画に適さない. このため, ここでは軌道を s に関して離散化し, 各離散点においてロボットと周辺障害物との交差判定を数値的に行う.

上述の問題設定では, ロボットの向き θ が拡張状態となる. 一方で, 移動パラメータ α は軌道の種類を表す離散値をとる. つまり, $\alpha \in \{F, FF, R, RR\}$ である. このように, 移動パラメータは最初から離散値であるので, 区間分割を考える必要はない.

図 6(a) に示すような駐車環境上に状態ロードマップを構築する. 図には位置ロードマップのノードとその接続関係も示している. 環境には三つの駐車スペースがあり, それぞれについて前進駐車用と後退駐車用の目標状態を表すノードを配置する. 移動ロボットのサイズは幅 1.0 [m], 全長 2.4 [m] とし, また最大曲率を $C^{\max} = 2.8^{-1} [\text{m}^{-1}]$ と

(a) 駐車環境と位置ロードマップ

(b) 状態ロードマップ (解像度レベル 5 のみ表示)

図 6 駐車環境と状態ロードマップ

(a) FF-R 軌道

(b) FF-R-F 軌道

図 7 異なる初期コンフィギュレーションからの駐車軌道

表 1 状態ロードマップのサイズ

resolution level	ノード数	横断遷移の数
1	90	0
2	174	0
3	342	16
4	678	1,067
5	1350	21,408

表 2 計算時間

	計算時間 [s]
状態ロードマップ構築	853.064
オンライン遷移生成	0.320
軌道計画	0.024

する. また, 状態ロードマップの最大解像度を $N_R = 5$ とする. 生成されたロードマップを図 6(b) に示す. 誌面の制約上, ここでは解像度レベル 5 のノードおよび横断遷移のみ示している. 各解像度レベルのノード数とリンク数を表 1 に示す.

異なる始点・終点に対して軌道計画を行った結果を図 7 に示す. 図 7(a) では, 位置 p_0 の近傍を始点とし, 終点を $(p_3^g, 1/2\pi)$ と設定した. その結果, ロボットはまず p_{13} まで前進し, 次に p_3^g へ後退して駐車する軌道が得られた. 図 7(b) では, ロボットは p_{16} へ前進し, p_8 へ後退してから p_4^g へ前進して駐車する軌道が得られた.

軌道生成の各段階に要した計算時間を表 2 に示す. 表中のオンライン遷移生成とは, 与えられた始点・目標点と状態ロードマップ中のノードとを接続する遷移をオンラインで生成する手続きを指す (第 2.3 節参照). 表に示された通り, 状態ロードマップ自体の生成には比較的大きな計算時間を要するが, 与えられた始点と終点に対する一時的なリンク生成や経路計画は短時間で完了していることが分かる.

図 8 小型電気自動車を用いた自動駐車

小型電気自動車を用いて自動駐車を行った。実験に用いた車両は与えられた軌道を追従して自動走行する機能を備えている。車両のサイズおよび曲率制約についてはシミュレーションと同様である。実験を行った環境用の状態ロードマップをオフラインで生成し、車載コントローラに記憶した。自己位置推定には両後輪に取り付けられたロータリエンコーダおよび前方に取り付けられた LIDAR からのセンサ情報をもとに拡張カルマンフィルタを用いた。初期位置から駐車スペースへの軌道を状態ロードマップを用いて生成し、これを目標軌道として軌道追従制御を行うことで自動的に駐車スペースへ走行した。実験の様子を図 8 に示す。後退→前進→後退の順序で切り替え、駐車スペースへの移動に成功した。今回は生成した軌道を単純なフィードバック制御により追従したため、その遅れのために最終的な駐車位置に若干の誤差が生じた。軌道から計算される操舵角情報を直接フィードフォワードすることにより、追従誤差の改善が見込まれる。

4. ま と め

ロボットの移動能力に関する制約を考慮した軌道計画に適した地図情報の生成法として、多分解能ロードマップ法の原理を述べ、その適用例を紹介した。本手法は移動モデルに関する二点境界値問題の解が与えられていれば適用可能であるので、潜在的な適用範囲は非常に広いと考えられる。例としては、無人航空機 (UAV) や 2 足歩行ロボットの倒立振子モデルなどへの適用が考えられる。また、オフライン手法の欠点として環境の変化に対する柔軟性が低いことが挙げられるが、準静的な障害物に対しては、その障害物によって遮蔽される遷移のみをロードマップから取り除いて計画を行うことで対応可能である。より基礎的な課題として、状態ロードマップの元となる位置ロードマップの設計があるが、これについては今のところ設計者の技巧

に頼るところが大きい。今後は障害物の配置や形状に応じて自動的に設計する手法の考案が求められる。

謝 辞 本稿の執筆にあたり、研究に従事した項警宇君、藤紘至君、また多くの有益な意見を頂戴した鈴木達也先生、ブレイン・レーブダール先生にこの場を借りて感謝申し上げる。

参 考 文 献

- [1] J.C. Latombe: Robot Motion Planning. Kluwer Academic, 1991.
- [2] H. Choset and J. Burdick: "Sensor Based Planning, Part I: The Generalized Voronoi Graph," IEEE International Conference on Robotics and Automation, pp.1649–1655, 1995.
- [3] L.E. Kavraki, P. Svetska, J.C. Latombe and M.H. Overmars: "Probabilistic Roadmaps for Path Planning in High-dimensional Configuration Spaces," IEEE Transactions on Robotics and Automation, vol.12, no.4, pp.566–580, 1996.
- [4] M.S. Branicky, S.M. LaValle, K. Olson and L. Yang: "Quasirandomized Path Planning," IEEE International Conference on Robotics and Automation, pp.1481–1487, 2001.
- [5] M. LaValle: "Randomized Kinodynamic Planning," International Journal of Robotics Research, vol.20, no.5, pp.378–400, 2001.
- [6] D. Hsu, R. Kindel, J.C. Latombe and S. Rock: "Randomized Kinodynamic Motion Planning with Moving Obstacles," International Journal of Robotics Research, vol.21, no.3, pp.233–255, 2002.
- [7] H. Noborio, T. Naniwa and S. Arimoto: "A Quadtree-based Path-planning Algorithm for a Mobile Robot," Journal of Field Robotics, vol.7, no.4, pp.555–574, 1990.
- [8] S. Behnke: 'Local Multiresolution Path Planning,' Robocup 2003: Robot Soccer World Cup VII. pp.332–343, Springer, 2004.
- [9] R.V. Cowlagi and P. Tsiotras: "Multiresolution Path Planning with Wavelets: A Local Replanning Approach," American Control Conference, pp.1220–1225, 2008.
- [10] J. Xiang, Y. Tazaki, T. Suzuki and B. Levedahl: "Variable-resolution Velocity-Time Roadmap Generation Considering Safety Constraints for Autonomous Vehicles," IEEE Conference on Decision and Control, pp.2281–2287, 2013.
- [11] 山田貴之, 項警宇, 田崎勇一, 鈴木達也: "可変分解能ロードマップによるマルチロボットのスケジューリングと軌道計画", ロボティクス・メカトロニクス講演会, 1A2-A06, 2013.
- [12] H. Fuji, J. Xiang, Y. Tazaki and T. Suzuki: "Trajectory Planning for Automated Parking Using Multi-resolution State Roadmap Considering Non-holonomic Constraints," IEEE Intelligent Vehicles Symposium, pp.407–413, 2014.

田崎勇一 (Yuichi Tazaki)

2008 年東京工業大学大学院情報理工学研究科情報環境学専攻博士課程修了。工学博士。2007 年～2009 年、日本学術振興会特別研究員。2008 年, Honda Research Institute Europe (Germany) 客員研究員。2009 年より名古屋大学大学院工学研究科助教, 現在に至る。ハイブリッドシステム理論や動的システムの離散抽象化, 多自由度ロボットの動作計画等に関する研究に従事。計測自動制御学会, IEEE 会員。(日本ロボット学会正会員)