三次元地形上における
二足歩行ロボットの
モデルベース脚配置計画

平成 27 年度

大阪市立大学大学院工学研究科
電子情報系専攻　前期博士課程

小　林　　大　気

概要

- My hovercraft is full of eels.
- A légpárnás hajóm tele van angolnákkal.
- 私のホバークラフトは鰻でいっぱいです。
- 小なり <; 等号 =; 大なり >;

二足歩行ロボットの自律移動には，目的地までの経路選択や軌道計画のみならず，局所的範囲での機構制約や地形に適応した，実時間での脚配置計画が重要となる．脚配置計画手法としてよく研究されている固定パターンの組み合わせ問題としてグラフ探索手法で解く探索ベース手法では，地形適応などが容易に行えるが，厳密な最適化はできず，探索における計算コストも高い．一方で，脚先遷移を運動学モデルで表現して逆運動学問題として解くモデルベースの手法も提案されており，原理的により厳密に最適化でき計算コストも低く抑えられるため，局所的な脚配置計画としてはより適していると考えられる．平面上では実時間で計画可能なモデルベース手法が既に提案されているが，三次元地形上での計画は実現されていなかった．

そこで，従来の水平方向移動のみを考慮した歩行モデルに独立した垂直方向の移動動作を加え，地形の形状情報に基づき，遊脚の接地点を支持脚基準の着地ワークスペース範囲内へ制限する制約を定式化し加えることで，実時間性を維持しつつ，三次元地形に適応可能なモデルベースの脚配置計画手法を提案した．二足歩行ロボット HRP-2 および NAO を想定した計画シミュレーションにより，段差など不連続な変化のある地形形状では最適化計算の不安定性に関して改善の余地があるが，連続的な形状の地形では安定して最適化された脚配置が得られており，またいずれにおいても実時間で計画できていることを示した．

目次

図目次

表目次

第 1 章

緒言

1.1 研究背景

ロボット技術の発展に伴い，ロボットの人間の生活環境への導入に対する期待も高まっている．それに伴い，人間の生活環境におけるロボットの動作に関する研究も多く行われ，作業代替・補助，移動能力など多岐にわたる．生活環境においてヒューマノイドロボットを運用するためには，様々な動作を行うためのベースとして，環境認識に基づく自律的な移動が必要となる．自律移動には，環境の認識，自己位置の推定，目標位置・姿勢（状態）までの歩行計画などが必要とされ，またリアルタイムに歩行を計画することが必要不可欠である．

二足歩行ロボットの自律移動には，目的地までの経路選択や軌道計画のみならず，局所的範囲での機構制約や地形に適応した，実時間での脚配置計画が重要となる．全方位移動車輪型ロボットの場合，目的地までの最小経路を直接使うことができる [1] が，その他のロボットの場合，それぞれの機構特性を考慮して計画する必要がある．例えば，非ホロノミック車輪型移動ロボットの場合，ハンドル操作などの非ホロノミック拘束 [2] を考慮しなければならない．二足歩行ロボットの場合，支持脚からみた遊脚の可動範囲，ロボットの安定性，歩行計画の最適性 [3–5] など，多くの制約条件を考慮しながら脚配置を計画する必要がある．小林，杉原らは足同士がぶつかったり交差することを避けるための脚配置決定法を提案している [6]．Chestnutt らは A*アルゴリズムを使った歩行計画を提案しており，ホンダの ASIMO で実験を行っている [7]．提案手法では，静的な物体だけでなく動的な障害物に対しても衝突を避ける歩行計画が可能となっている．Hornung らは D* Lite アルゴリズムによる探索手法で歩行計画を行う手法を提案している [8]．提案手法はヒューマノイドロボット NAO に実装され，動的に変化する環境中での歩行計画が可能であることが示されている．また環境の複雑さに合わせて経路のみの計画と脚配置の計画とを切り替えつ歩行計画を行う手法も提案している [9]．Liu らは，RRT を用いて経路を生成しつつ，用意した固定の踏み

出しパターンとその配置を微調整するパターンとを環境に合わせて展開していくことで計画する手法を提案している [11]．Huang らはエネルギ消費に着目して歩行計画を最適にする手法を提案している [10]．提案手法は，歩幅や旋回角で決定される消費されるエネルギを最適化するために A* アルゴリズムを使った探索手法によりエネルギ効率を最適にする歩行計画を決定している．これらのグラフ探索などに基づいた探索ベースの計画手法では，ハードウェアの制約や地形適応などはグラフ上でのノード展開の可否の判断だけで表現できるため容易に表現できるが，原理的に離散的な組み合わせの中の最適化であるため，連続値空間上での厳密な最適化にはならず，また効率化を図っても多数のノード展開による計算コストの大きさは無視できない．一方，モデルベースの歩行計画手法として，Kanoun らにより脚の接地位置を連結したチェインモデルによる歩行計画手法が提案されている [12, 13]．提案手法は，冗長マニピュレータの逆運動学モデルを解く手法と同様のアルゴリズムを用いることが可能であり，歩行後にマニピュレータにより対象物を把持する動作計画を実現した．筆者らは，二次元平面上において，二足歩行ロボットの脚配置をピボッティング操作と伸縮操作の両方が可能な幾何学的な二足歩行モデルに当てはめることで，運動計画を離散時間非ホロノミック系に適用する手法を提案してきた [16, 17]．提案手法は，二足歩行ロボットの脚位置を決定するモデルと倒立振子モデルにより計算されるエネルギ効率を考慮することで目標位置へ到達するための歩容計画の最適性が直接求められる利点がある．また，目標変更に対してリアルタイムで再計画を行うことができるなど，実時間性もある手法である．これらの研究の多くは二次元平面内での歩行計画を取り扱っており，三次元の地形を考慮したモデルベースの歩行計画手法の提案は少ない．Diets らは，障害物を含む三次元マップ上において，マップを障害物のない凸領域に可能な限り大きい単位で分割し [15]，凸領域の選択と脚配置の遷移の計画を整数混合最適化問題として解く [14] という三次元のモデルベース計画手法を提案しているが，地形によっては 10–20 歩程度の計画で 1 分近く計算時間がかかるなど，実時間性に課題が残っている．

1.2 提案

そこで本論文では，[17] の手法を拡張し，Fig. 1.1 に示すように三次元の形状を持った地形上においてモデルの制約条件に沿って脚配置計画を行う手法を提案する．三次元の地形は事前に外界センサなどの計測によりメッシュ状の地形データとして取得できている状態であると仮定する．そのデータに基づき，脚の接地条件を制約として表現して追加するのみで，実時間性を保ったまま三次元での計画を実現する．歩行モデル上の可動領域の限界や地形との接地条件を厳密に考慮するために制約条件付きの二次計画法を利用した反復法により最適な脚配置を得る手法を適用し，ヒューマノイドロボット HRP-2 [23] を想定した計画シミュレーション，および NAO [24] に基づく計画シミュレーションと実機実験により，提案手法

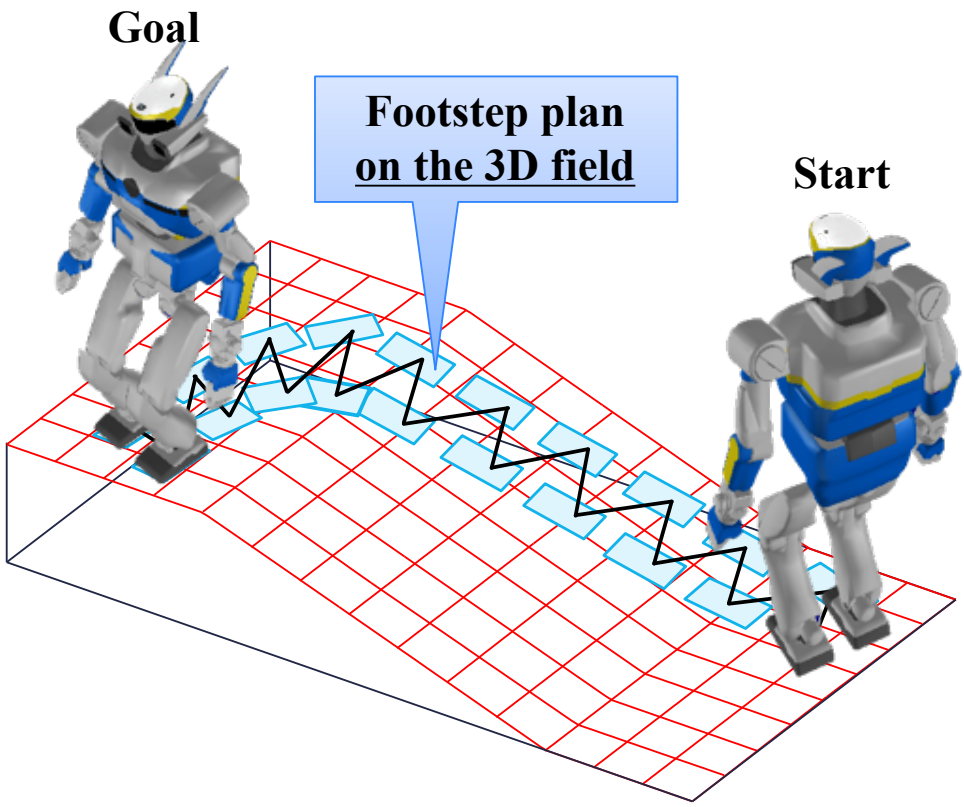


Fig. 1.1　3D footstep plan (on the slope way)

の有効性を示す．

1.3　本論文の構成

本論文の構成は，次のようになっている．まず本章において，本研究の背景・目的などを述べた．2 章では，脚配置計画のための二足歩行ロボットのモデルと計画問題の定式化について述べる．3 章では，定式化した問題を解くための最適化計算の手法と全体のアルゴリズムについて述べる．4 章では，様々な地形上での脚配置計画を行った結果とそれについての考察を述べる．5 章では，計画によって得られた脚配置を用いた実機による歩行実験とその結果について述べる．最後に 5 章では，本研究の結果などについてまとめ，今後の課題を述べる．

第 2 章

脚配置計画問題の定式化と三次元適応

2.1 はじめに

本章では，脚配置計画を実現するための提案するモデルベース計画手法における，歩行のモデル化から計画問題の定式化，および従来の平面上の計画手法の三次元適応のための拡張方法について記述する．また，本論文で用いる最適化問題を解くための計算手法と定式化についても述べる．

2.2 導入モデル

二足歩行ロボットの歩行動作モデルとして，Fig. 2.1 に示す簡易的な運動学モデルを導入する．このモデルでは，ピボット操作および両脚間隔の伸縮操作を反復することによって歩行を表現する．平面上の脚配置は，Fig. 2.2 に示すように脚先の幾何学的な配置関係のモデルで表される．脚先形状は矩形で考える．(x^i, y^i) はワールド座標系 Σ_0 における左脚の脚先中心の位置座標である．θ^i は Σ_0 の x 軸と両脚先間を結ぶ長さ l^i の線分とがなす角である．左右の脚先座標系 Σ_{fl}，Σ_{fr} は脚先の前方向に x 軸，内側の方向に y 軸をとる．Σ_0 の y 軸と Σ_{fl} および Σ_{fr} の x 軸とがなす角をそれぞれ θ^i_{fl}，θ^i_{fr} とする．なお，$i = 0, 1, \ldots, k-1$ (k は後述のステップ数)．

歩行動作は，はじめに右脚を軸として u^i_A 回転し，また両脚間隔 l^i を u^i_{l1} 伸ばし，さらに脚先を θ^i_{fl} から u^i_{fl} だけ回転させて θ^{i+1}_{fl} として着地する．次いで左脚を軸に u^i_B 回転し,u^i_{l2} だけ伸ばして l^{i+1} とし，脚先を θ^i_{fr} から u^i_{fr} だけ回転させて θ^{i+1}_{fr} として着地する．この一

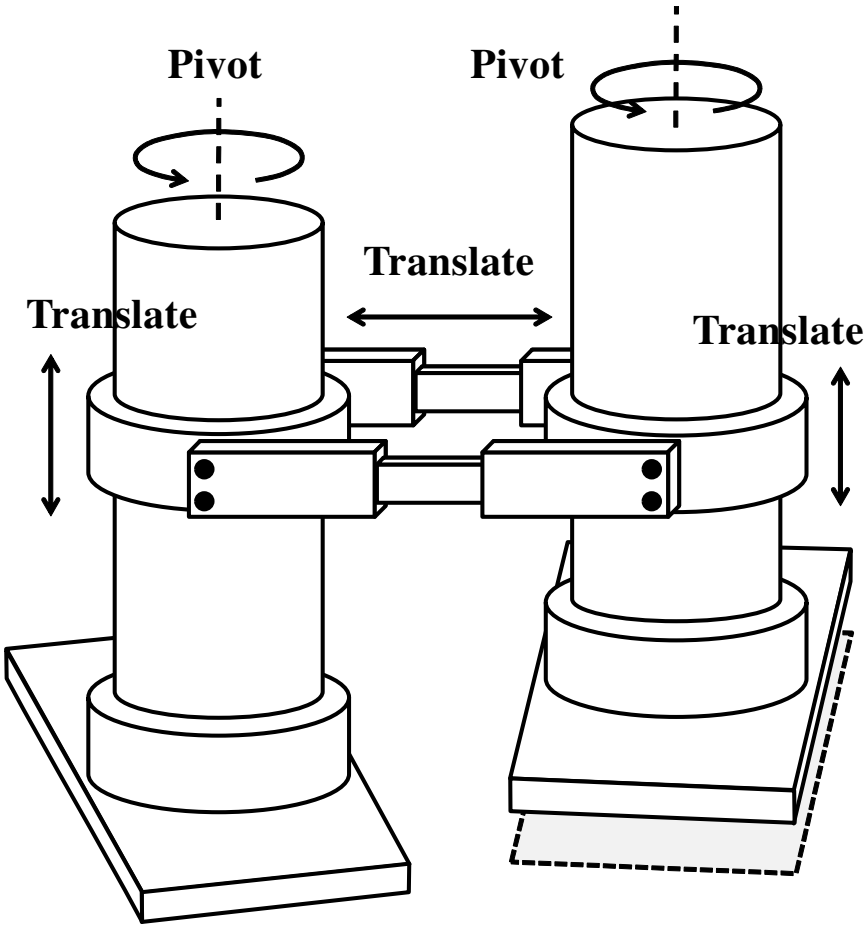


Fig. 2.1　Biped walking model

連の動作を 1 ステップとする．この 1 ステップの動作は，式 (2.1) に示す 6 状態 6 入力の状態方程式として表される．

$$
\begin{aligned}
x^{i+1} &= x^i + l^i \cos\theta^i - \left(l^i + u_{l1}^i\right) \cos\left(u_A^i - \theta^i\right) \\
y^{i+1} &= y^i + l^i \sin\theta^i + \left(l^i + u_{l1}^i\right) \sin\left(u_A^i - \theta^i\right) \\
\theta^{i+1} &= \theta^i - u_A^i + u_B^i \\
l^{i+1} &= l^i + u_{l1}^i + u_{l2}^i \\
\theta_{fl}^{i+1} &= \theta_{fl}^i + u_{fl}^i \\
\theta_{fr}^{i+1} &= \theta_{fr}^i + u_{fr}^i
\end{aligned}
\tag{2.1}
$$

このモデルの状態変数は $\boldsymbol{q}^i = (x^i,\, y^i,\, \theta^i,\, l^i,\, \theta_{fl}^i,\, \theta_{fr}^i)$, 入力変数は $\boldsymbol{u}^i = (u_A^i,\, u_B^i,\, u_{l1}^i,\, u_{l2}^i,\, u_{fl}^i,\, u_{fr}^i)$

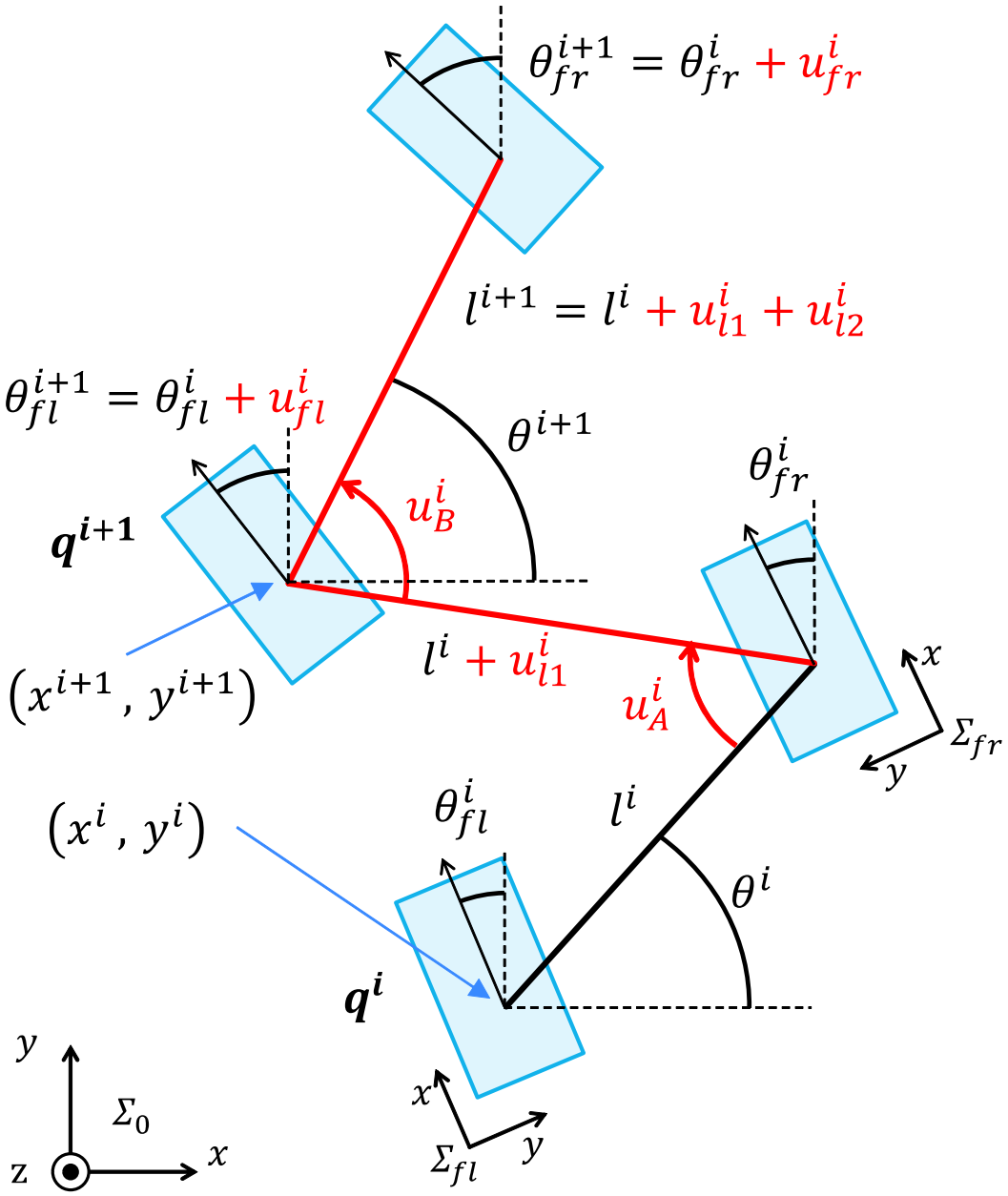


Fig. 2.2　Foothold model

と定義される．また，状態方程式式 (2.1) は $\boldsymbol{q}^i$，$\boldsymbol{u}^i$ を用いて式 (2.2) で表される．

$$\boldsymbol{q}^{i+1} = \boldsymbol{G}\left(\boldsymbol{q}^i, \boldsymbol{u}^i\right) \tag{2.2}$$

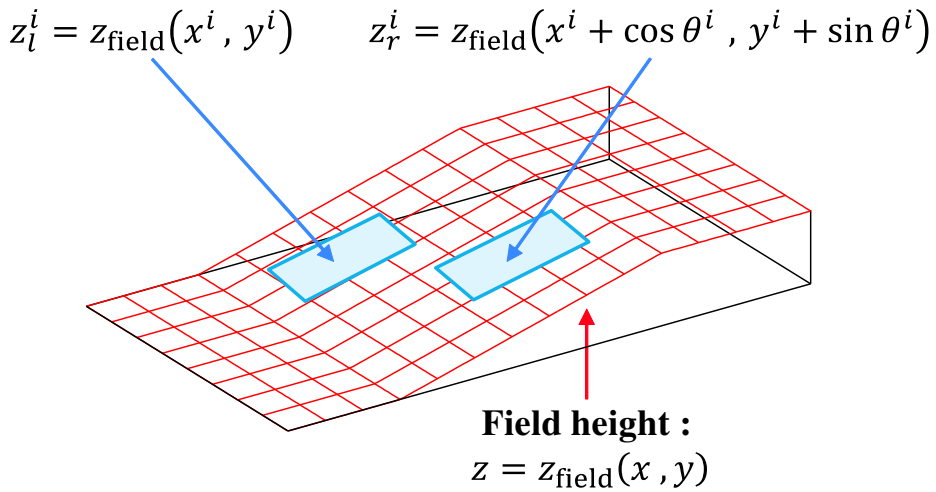


Fig. 2.3　Field map model

2.3　脚先の高さの取り扱い

前節で述べた脚配置モデル式 (2.1) は平面上の遷移についてのものであり，$\boldsymbol{q}^i$，$\boldsymbol{u}^i$ は要素に脚先高さに関するものを含まない．これは，提案する運動学モデルが，脚の垂直方向の移動を平面上の動作と独立して行えるという仮定のもとで構築されていることに由来する．地面への接地を考えると，脚先の z 座標は xy 平面上の位置における地形形状に合わせることになる．左右の脚先の xy 平面上の座標それぞれを $(x_l^i, y_l^i) = (x^i, y^i)$，$(x_r^i, y_r^i) = (x^i + l^i \cos\theta^i, y^i + l^i \sin\theta^i)$ とすれば，z_l^i，z_r^i は Fig. 2.3 に示すように式 (2.3) で表すことができる．

$$\begin{aligned} z_l^i &= z_{\text{field}}\left(x_l^i, y_l^i\right) \\ z_r^i &= z_{\text{field}}\left(x_r^i, y_r^i\right) \end{aligned} \tag{2.3}$$

地形形状 $z_{\text{field}}(x, y)$ が既知であれば，z_l^i，z_r^i は式 (2.3) を用いて $\boldsymbol{q}^i$ で表現できる．これを用いて地形との接触制約を課すことにより，平面上の脚配置モデルでありながら地形を考慮した三次元の脚配置計画を行うことが可能となる．具体的な制約式については 2.6 節で示す．

2.4　計画問題の定式化と解法

式 (2.1) より，k ステップ後の状態 $\boldsymbol{q}^k$ は初期状態 $\boldsymbol{q}^0$ と入力列 $\boldsymbol{U}_k = ({\boldsymbol{u}^0}^T, \ldots, {\boldsymbol{u}^{k-1}}^T)^T$ を用いて $\boldsymbol{q}^k = \boldsymbol{G}_k(\boldsymbol{q}^0, \boldsymbol{U}_k)$ と表すことができる．目標状態 $\boldsymbol{q}_{\text{goal}}$ へ k ステップで到達するとしたときの脚配置計画問題は，式 (2.4) の非線形代数方程式の解 $\boldsymbol{U}_k$ を求める逆運動学問題

となる．

$$\boldsymbol{q}_{\text{goal}} = \boldsymbol{G}_k\left(\boldsymbol{q}^0, \boldsymbol{U}_k\right) \tag{2.4}$$

この問題式 (2.4) を $\boldsymbol{U}_k$ について解けば脚配置計画が求まる．

本論文では，目標追従を評価関数で表し，制約条件下での最適化問題として解くことを考える．$\boldsymbol{U}_k$，$\boldsymbol{q}^0$ を変数とする評価関数を $V = H(\boldsymbol{q}^0, \boldsymbol{U}_k)$，制約条件を $\boldsymbol{\omega}(\boldsymbol{q}^0, \boldsymbol{U}_k) \geq \boldsymbol{0}$ として，式 (2.5) の最適化問題を定義する．

$$\begin{aligned} &\text{minimize} \quad V = H\left(\boldsymbol{q}^0, \boldsymbol{U}_k\right) \\ &\text{subject to} \quad \boldsymbol{\omega}\left(\boldsymbol{q}^0, \boldsymbol{U}_k\right) \geq \boldsymbol{0} \end{aligned} \tag{2.5}$$

この式 (2.5) の非線形計画問題を，反復法を用いて解くことを考える．反復のある段階での入力列 $\boldsymbol{U}_k$ に微小変化 $\boldsymbol{\Delta U}_k$ を加えて $\boldsymbol{U}_k \leftarrow \boldsymbol{U}_k + \boldsymbol{\Delta U}_k$ として更新するとき，更新前の評価値 V_{now} は，更新後の評価値 V_{next} と変化量 ΔV を用いて $V_{\text{next}} = V_{\text{now}} + \Delta V$ と表すことができる．線形近似を用いて V_{next} を $\boldsymbol{\Delta U}_k$ に関する二次形式，制約条件を一次式に近似し，更新量 $\boldsymbol{\Delta U}_k$ を求める問題を式 (2.6) の制約条件付きの二次計画問題に定式化する．

$$\begin{aligned} &\text{minimize} \quad \Delta V = \frac{1}{2}\boldsymbol{\Delta U}_k^T \boldsymbol{\Phi}\, \boldsymbol{\Delta U}_k + \boldsymbol{\phi}^T \boldsymbol{\Delta U}_k \\ &\text{subject to} \quad \boldsymbol{\Omega}\, \boldsymbol{\Delta U}_k + \boldsymbol{\omega} \geq \boldsymbol{0} \end{aligned} \tag{2.6}$$

この問題式 (2.6) を逐次解いて $\boldsymbol{U}_k$ を更新していくことで，計画問題の解が得られる．

なお, 以降では入力列更新前と後の変数をそれぞれ $(\text{variable\_name})_{\text{now}}$, $(\text{variable\_name})_{\text{next}}$ の形で表すこととする．

2.5 評価関数

本論文では，目標状態到達を計画の指標とし，最適化する評価関数には目標追従二乗誤差を用いる．

$$V = \frac{1}{2}\|\boldsymbol{\varepsilon}\|^2 = \frac{1}{2}\left\|\boldsymbol{q}_{\text{goal}} - \boldsymbol{q}^k\right\|_{K_p}^2 \tag{2.7}$$

K_p は重み行列である．これを式 (2.5) の問題の評価関数として設定する．なお，目標追従以外の指標（ [16,17] における運動エネルギなど）も評価関数に加える事で最適化可能である．

問題式 (2.6) へ適用するため，$\boldsymbol{\Delta U}_k$ によって更新された後の状態 $\boldsymbol{q}_{\text{next}}^k$ を，$\boldsymbol{q}_{\text{now}}^k$ の $\boldsymbol{U}_k$ についてのヤコビ行列 $J_k = \partial \boldsymbol{G}_k / \partial \boldsymbol{U_k}$ を用いて線形近似する．

$$\boldsymbol{q}_{\text{next}}^k = \boldsymbol{q}_{\text{now}}^k + J_k\, \boldsymbol{\Delta U}_k \tag{2.8}$$

式 (2.8) を用いて二次計画問題式 (2.6) の評価関数を式 (2.9) で定義する．

$$
\begin{aligned}
V_{\text{next}} &= \frac{1}{2} \left\| \varepsilon_{\text{next}} \right\|_{K_p}^2 + \frac{1}{2} \left\| \boldsymbol{\Delta U}_k \right\|_{K_u}^2 + \frac{1}{2} \left\| \boldsymbol{\Delta U}_{zk} \right\|_{K_{uz}}^2 \\
&= \frac{1}{2} \left(\varepsilon_{\text{now}} - J_k \, \boldsymbol{\Delta U}_k \right)^T K_p \left(\varepsilon_{\text{now}} - J_k \, \boldsymbol{\Delta U}_k \right) \\
&\quad + \frac{1}{2} \, \boldsymbol{\Delta U}_k^T K_u \, \boldsymbol{\Delta U}_k \\
&\quad + \frac{1}{2} \left(J_{zk} \, \boldsymbol{\Delta U}_k \right)^T K_{u_z} \left(J_{zk} \, \boldsymbol{\Delta U}_k \right) \\
&= \frac{1}{2} \, \boldsymbol{\Delta U}_k^T \boldsymbol{\Phi} \, \boldsymbol{\Delta U}_k + \boldsymbol{\phi}^T \boldsymbol{\Delta U}_k + \text{const.} \\
\boldsymbol{\Phi} &= J_k^T \, K_p J_k + K_u + J_{zk}^T \, K_{u_z} J_{zk} \\
\boldsymbol{\phi} &= -J_k^T \, K_p \, \varepsilon_{\text{now}}
\end{aligned}
\tag{2.9}
$$

K_p，K_u，K_{u_z} は評価関数の各項を重み付ける対角行列である（対角成分の設定方法は 3.3 節に示す）．$\boldsymbol{\phi} \in R^{6k}$ はベクトル，$\boldsymbol{\Phi} \in R^{6k \times 6k}$ は実対称の正定値行列である．

式 (2.9) の式変形前の V_{next} の定義において，第一項は前述の目標追従二乗誤差の評価項，第二項は入力列の更新量 $\boldsymbol{\Delta U}_k$ 最小化の評価項である．第三項は，高さ変位を最小化するための項である．脚配置モデルが状態として z_l^i，z_r^i を持つと仮定すると，式 (2.10) のように次ステップへ遷移すると考えられる．

$$
\begin{aligned}
z_l^{i+1} &= z_l^i + u_{zl}^i \\
z_r^{i+1} &= z_r^i + u_{zr}^i
\end{aligned}
\tag{2.10}
$$

高さに対する仮想的な入力 u_{zl}^i，u_{zr}^i は式 (2.3) を用いて $\boldsymbol{q}^i$，$\boldsymbol{q}^{i+1}$，$z_{\text{field}}(x, y)$ から計算でき，高さに対する仮想入力列 $\boldsymbol{U}_{zk} = (u_{zl}^0, u_{zr}^0, \ldots, u_{zl}^{k-1}, u_{zr}^{k-1})^T$ は $\boldsymbol{q}^0$，$\boldsymbol{U}_k$ で表すことができる．入力列更新時の $\boldsymbol{U}_{zk}$ の変化 $\boldsymbol{\Delta U}_{zk}$ を最小化すれば，高さ方向の変位を最小化する計画が得られる．なお，式 (2.9) 中の J_{zk} は，$\boldsymbol{U}_{zk}$ の $\boldsymbol{U}_k$ についてのヤコビ行列 $J_{zk} = \partial \boldsymbol{U}_{zk} / \partial \boldsymbol{U}_k$ である．

2.6 制約条件

制約条件は，三次元上の脚配置を実現するために必要となる地形との接触についての制約と，適用するロボットに合わせて定式化されるハードウェア上の制約からなる．

2.6.1 地形との接触制約

平面上で定義されたモデルを用いて三次元の脚配置を実現するため，地形との接触について制約を課す必要がある．脚の垂直方向の移動が平面動作から独立していること，機構的に踏み出し時の高さには限界があることを踏まえ，式 (2.3) を用いて両脚間の高低差について制約を課す．

$$\begin{aligned} -z_{\max} \leq z_l^{i+1} - z_r^i \leq z_{\max} \\ -z_{\max} \leq z_r^{i+1} - z_l^{i+1} \leq z_{\max} \end{aligned} \tag{2.11}$$

$z_{\max}$ は高低差の絶対値上限である．この制約を課すと，歩行時の垂直方向の移動が上下限値内に収まるように歩幅を抑えた脚配置が求まることになり，地形を考慮した脚配置計画が実現される．地形形状に関しては，平面位置に対応する高さ $z_{\text{field}}(x, y)$，線形近似時に必要となる x，y 方向それぞれの傾き $\partial z_{\text{field}}/\partial x$, $\partial z_{\text{field}}/\partial y$ が既知であればよい．上限値 $z_{\max}$ は，定数としても関数としてもよい．即ち，踏み出しの高さによって遊脚の水平方向の可動範囲が変化するような場合には，$\boldsymbol{q}^i$，$\boldsymbol{u}^i$ および z_{field} を用いて表現される関数とするなど，適用するロボットに合わせて定式化を行えばよい．

なお，現在は地形の傾斜に対する脚先の姿勢を考慮していない．今後，脚裏形状や制御における許容範囲，地形形状情報から定式化できると考えている．

2.6.2 遊脚の配置制約

制約条件を定式化するために，Fig. 2.4，Fig. 2.5 および Fig. 2.6 中に示している，次の三つのパラメータを定義する．

- 支持脚と遊脚の相対距離 L^i
- 支持脚の y 軸から見た直線（L^i）の傾き θ_1^i
- 支持脚を中心としたときの両脚先間の相対角度 θ_2^i

L^i，θ_1^i，θ_2^i はそれぞれ，右脚支持の場合に $L^i = l^i + u_{l1}^i$，$\theta_1^i = u_A^i - \theta^i + \theta_{fr}^i$，$\theta_2^i = \theta_{fl}^{i+1} - \theta_{fr}^i$，左脚支持の場合に $L^i = l^{i+1}$，$\theta_1^i = \theta^{i+1} - \theta_{fl}^{i+1}$，$\theta_2^i = \theta_{fl}^{i+1} - \theta_{fr}^{i+1}$ となる．

まず，支持脚に対する遊脚の相対角度 θ_2^i に制約を課す．これは式 (2.12) で定式化される（Fig. 2.4 参照）．

$$\theta_{2\min} \leq \theta_2^i \leq \theta_{2\max} \tag{2.12}$$

$\theta_{2\max}$，$\theta_{2\min}$ は相対角度の上下限値である．

次に，支持脚に対する遊脚の可動範囲を Fig. 2.5 に示す領域内に制限する制約を課す．支

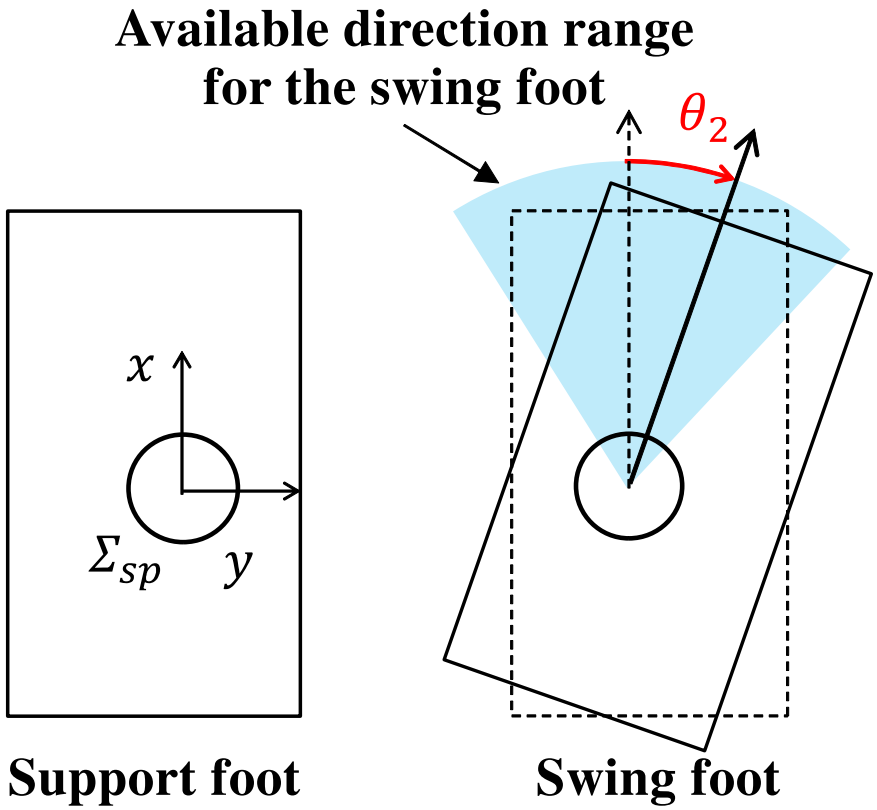


Fig. 2.4　Swinging foot workspace of the relative angle between the left and the right leg

持脚に対する遊脚の相対位置座標 $(x_{sw}^i, y_{sw}^i) = (L^i \sin\theta_1^i, L^i \cos\theta_1^i)$ は，$x_{sw}^i \geq 0$，$x_{sw}^i < 0$ それぞれの範囲において，$(0, Y_{\min})$ を中心とする楕円形の領域に制限されることとなる．この楕円の径については，$x_{sw}^i \geq 0$，$x_{sw}^i < 0$ それぞれの領域において，長軸は共通だが短軸は異なる．

$$
\begin{gathered}
y_{sw}^i \geq Y_{\min} \\
\left(\frac{x_{sw}^i}{X}\right)^2 + \left(\frac{y_{sw}^i - Y_{\min}}{Y_{\max} - Y_{\min}}\right)^2 \leq 1.0 \\
\left(X = \begin{cases} X_{\max} & (x_{sw}^i \geq 0) \\ X_{\min} & (x_{sw}^i < 0) \end{cases}\right)
\end{gathered}
\tag{2.13}
$$

$Y_{\max}$，$Y_{\min}$ は Σ_{sp} の y 軸上の両脚間隔の上限値及び下限値．また，$X_{\max}$，$X_{\min}$ は，脚間距離が最小のときの前進・後進それぞれについての Σ_{sp} の x 軸方向の歩幅上限値．

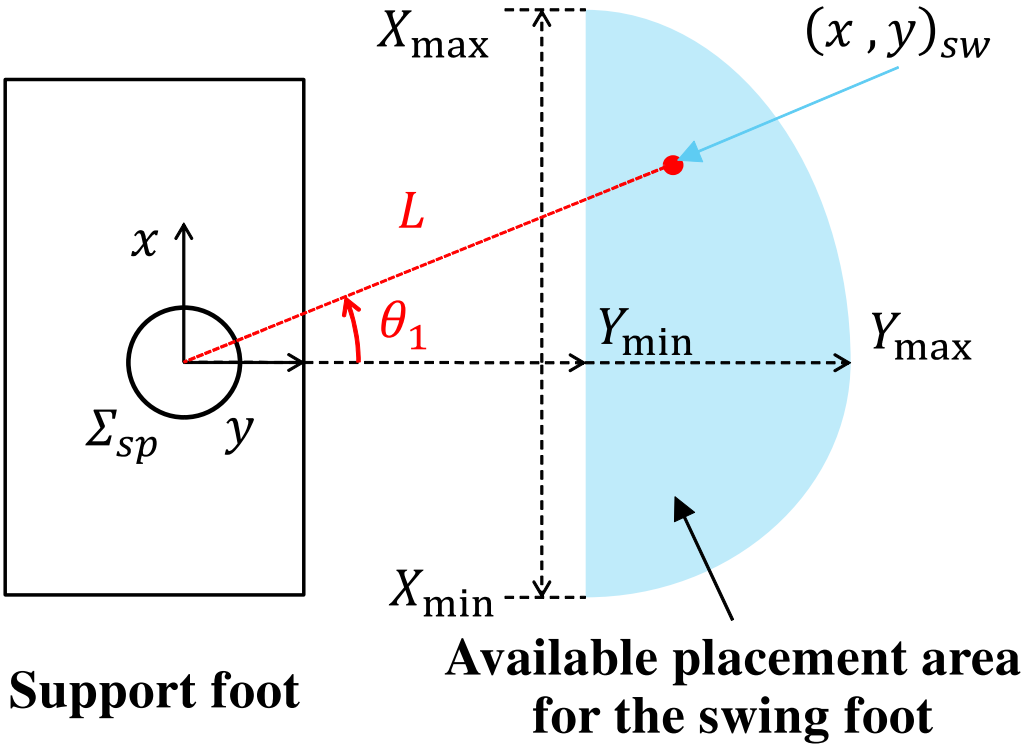


Fig. 2.5　Swinging foot workspace of the placement between the left and the right leg

2.6.3　脚先の干渉防止制約

さらに，脚先が重なることを防ぐための制約を課す．Fig. 2.6 に示している，支持脚座標系 Σ_{sp} から見た遊脚の頂点 $\boldsymbol{R}_A^i = (x_{R_A}^i, y_{R_A}^i)$ と $\boldsymbol{R}_B^i = (x_{R_B}^i, y_{R_B}^i)$，また遊脚座標系 Σ_{sw} から見た左脚の頂点 $\boldsymbol{L}_A^i = (x_{L_A}^i, y_{L_A}^i)$ と $\boldsymbol{L}_B^i = (x_{L_B}^i, y_{L_B}^i)$ は式 (2.14) で表される．

$$
\begin{aligned}
x_{R_A}^i &= L^i \sin\theta_1^i + O_{xf} \cos\theta_2^i + O_{yi} \sin\theta_2^i \\
y_{R_A}^i &= L^i \cos\theta_1^i + O_{xf} \sin\theta_2^i - O_{yi} \cos\theta_2^i \\
x_{R_B}^i &= L^i \sin\theta_1^i - O_{xb} \cos\theta_2^i + O_{yi} \sin\theta_2^i \\
y_{R_B}^i &= L^i \cos\theta_1^i - O_{xb} \sin\theta_2^i -^i O_{yi} \cos\theta_2^i \\
x_{L_A}^i &= L^i \sin\left(-\theta_1^i - \theta_2^i\right) + O_{xf} \cos\theta_2^i + O_{yi} \sin\theta_2^i \\
y_{L_A}^i &= L^i \cos\left(-\theta_1^i - \theta_2^i\right) + O_{xf} \sin\theta_2^i - O_{yi} \cos\theta_2^i \\
x_{L_B}^i &= L^i \sin\left(-\theta_1^i - \theta_2^i\right) - O_{xb} \cos\theta_2^i + O_{yi} \sin\theta_2^i \\
y_{L_B}^i &= L^i \cos\left(-\theta_1^i - \theta_2^i\right) - O_{xb} \sin\theta_2^i - O_{yi} \cos\theta_2^i
\end{aligned}
\tag{2.14}
$$

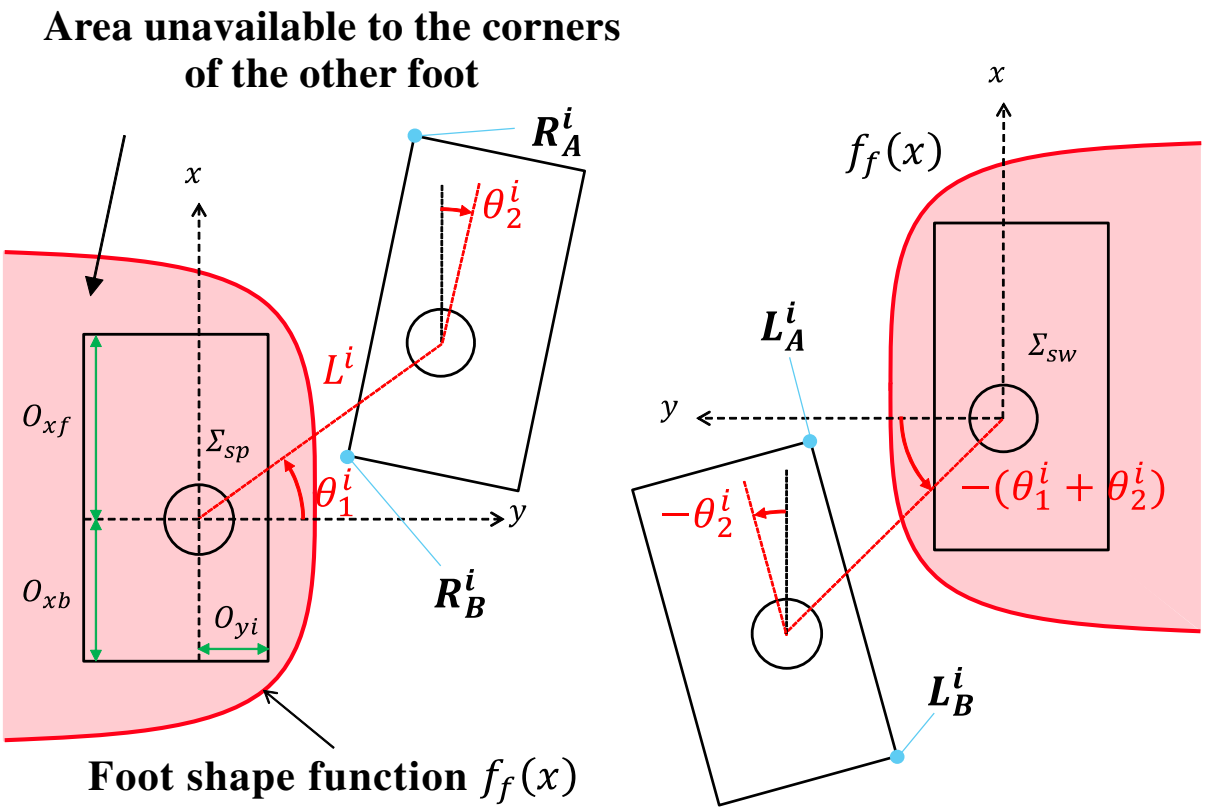


Fig. 2.6　Constraints to prevent the overlap of feet

提案するモデルにおける脚先形状は実際のロボットの脚先を包含できる最小の長方形としている．$O_{yi}O_{yo}$ はそれぞれ脚先中心から内側および外側の辺までの長さ，O_{xf}，O_{xb} はそれぞれ脚先中心から前端，後端までの長さである．また，入力列の更新前後で制約式が不連続に変化するのを防ぐために，Fig. 2.7 に示すような左脚座標系から見た左脚内側，右脚座標系から見た右脚内側の形状を表現する関数 $f_f(x)$ を導入する．脚先座標系の y 軸の正の方向に凸で，パラメータによって矩形に近づけられる関数が望ましく，本論文では三つのパラメータ a，b，c をもつ x の 10 乗の関数とした．

$$f_f(x) = a(x-b)^{10} + c \tag{2.15}$$

パラメータは Levenberg-Marquardt 法による非線形最小二乗推定によりフィッティングを

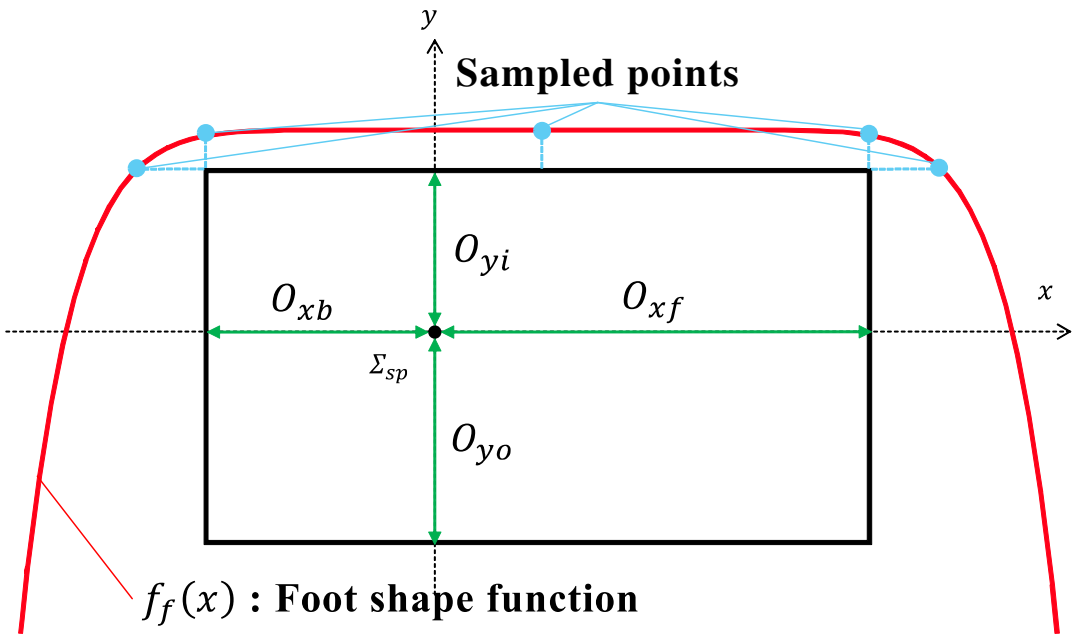


Fig. 2.7　Foot shape function

行った．これを用いて，制約条件を式 (2.16) で表現する．

$$\begin{aligned} f_f\left(x^i_{R_A}\right) \leq y^i_{R_A} \quad , \quad f_f\left(x^i_{R_B}\right) \leq y^i_{R_B}, \\ f_f\left(x^i_{L_A}\right) \leq y^i_{L_A} \quad , \quad f_f\left(x^i_{L_B}\right) \leq y^i_{L_B} \end{aligned} \tag{2.16}$$

2.6.4　二次計画問題への適用

以上の制約条件 $\omega_i(\boldsymbol{q}^0, \boldsymbol{U}_k) \geq 0$ $(i = 0, \ldots, 20k-1)$ は，まとめて $\boldsymbol{\omega}(\boldsymbol{q}^0, \boldsymbol{U}_k) = (\omega_0, \ldots, \omega_{20k-1})^T \geq \boldsymbol{0}$ $(\boldsymbol{\omega} \in R^{20k})$ と表される．これを問題式 (2.6) に適用するため，更新後の制約条件について線形近似を行う．

$$\boldsymbol{\omega}_{\text{next}} = \boldsymbol{\omega}_{\text{now}} + \Omega\, \boldsymbol{\Delta U_k} \geq \boldsymbol{0} \tag{2.17}$$

$\Omega \in R^{20k \times 6k}$ は $\boldsymbol{\omega}_{\text{now}}$ の $\boldsymbol{U}_k$ についてのヤコビ行列 $\Omega = \partial \boldsymbol{\omega}_{\text{now}} / \partial \boldsymbol{U}_k$ である．

2.7　本章のまとめ

本章では，三次元適応のための歩行モデルの拡張および追加の制約条件について説明した．従来の水平方向の脚配置計画のためのモデルに，独立した垂直方向の動作を追加し，脚

配置遷移モデルの状態および入力についての次元の拡張を行うことなく，事前に与えられた地形情報に基いて垂直方向の移動量を制限する制約を定式化し追加することで，計算コスト増加を抑えつつ三次元地形への適応を可能にした．次章では，本章で定義した計画問題の解法について，全体のアルゴリズムおよび求解性能の安定化のためのいくつかの手法について述べる．

第 3 章

脚配置計画アルゴリズム

3.1 はじめに

本章では，前章で定式化した問題を解くためのアルゴリズムについて説明する．反復法を用いた計算アルゴリズムによって計算を行い，脚配置を計算する．即ち，定式化した二次計画問題式 (2.6) を逐次解き，入力列 $\boldsymbol{U}_k$ を更新して解に近づけていくという計算を行う．

全体の計画のアルゴリズムを Fig. 3.1 に示す．各要素について述べていく．

3.2 初期ステップ数決定と入力列初期化

まずはじめに，計画開始時に用いる初期ステップ数 k_{init} を設定する．平面上と高さ方向それぞれについて，目標位置までの距離を 1 ステップあたりの最大の直進距離 $d_{xy\,\max}$，$d_{z\,\max} = 2\,z_{\max}$ で割ることで所要ステップ数を求め（平面上の最大直進距離は，事前に直線上を進む計画問題を解いておき，得られた解の直進距離を用いる），二つのうち大きい方を採用する．さらに，これに実用上 1 を加えたものを k_{init} として設定する．これは，先の計算で求めたものはあくまでも直進で目標に向かうことができる場合の最小値であり，地形や初期・目標状態の姿勢を考慮すると実際には求まったステップ数では到達できないと考えられるからである．

$$
\begin{aligned}
k_{\text{init}} &= \left\lceil \max\left(k_{xy\max},\, k_{z\max}\right) \right\rceil + 1 \\
k_{xy\max} &= \frac{\left\| \left(x_{\text{goal}}, y_{\text{goal}}\right) - \left(x^0, y^0\right) \right\|}{d_{xy\,\max}} \\
k_{z\max} &= \max\left(\frac{\left\| z_{l\text{goal}} - z_l^0 \right\|}{d_{z\,\max}},\, \frac{\left\| z_{r\text{goal}} - z_r^0 \right\|}{d_{z\,\max}} \right)
\end{aligned}
\tag{3.1}
$$

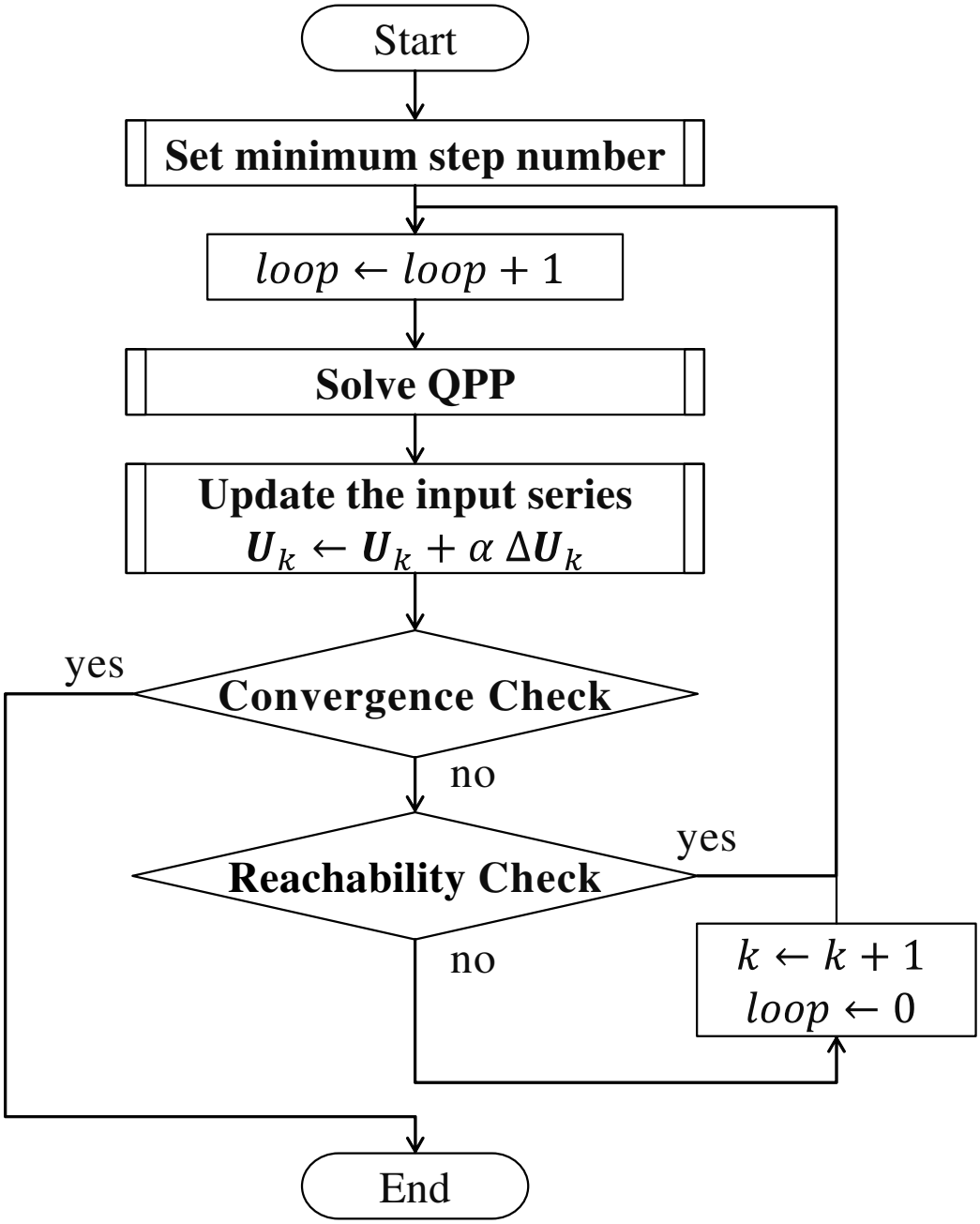


Fig. 3.1　Footstep planning algorithm

また，入力列は零ベクトルで初期化する．

3.3 二次計画問題の求解と入力列更新

3.3.1 Goldfarb-Idnani 法による二次計画問題の求解

“Solve QPP” では，各反復における計画問題式 (2.9) を解き $\boldsymbol{\Delta U}_k$ を求める．解法としては，式 (2.9) が凸二次計画問題であることから，有効制約法の一種である Goldfarb-Idnani (GI) 法 [18] を用いる．

線形近似により実行不可能な問題となってしまった場合は，若干の制約違反を許す緩和問題を設定して解く．文献 [19] で示されている無限大ノルムを用いた緩和問題式 (3.2) を設定する．

$$\begin{aligned} &\text{minimize} \quad \frac{1}{2}\,\boldsymbol{\Delta U}_k^T\,\boldsymbol{\Phi}\,\boldsymbol{\Delta U}_k + \boldsymbol{\phi}^T\,\boldsymbol{\Delta U}_k + \rho\,\xi \\ &\text{subject to} \quad \boldsymbol{\Omega}\,\boldsymbol{\Delta U}_k + \boldsymbol{\omega} + \xi\,\mathbf{1} \geq \mathbf{0}, \quad \xi \geq 0 \end{aligned} \tag{3.2}$$

$\rho > 0$ はペナルティパラメータ，ξ はスラック変数，$\mathbf{1}$ は全要素が 1 のベクトルである．この問題はすでに二次計画問題ではなくなっているため，式 (3.3) および式 (3.3) に示すように二次計画問題へ近似する．

$$\begin{aligned} &\text{minimize} \quad \frac{1}{2}\,\boldsymbol{\Delta U}_k^T\,\boldsymbol{\Phi}'\,\boldsymbol{\Delta U}_k + \boldsymbol{\phi}'^T\,\boldsymbol{\Delta U}_k \\ &\text{subject to} \quad \boldsymbol{\Omega}'\,\boldsymbol{\Delta U}_k + \boldsymbol{\omega}' \geq \mathbf{0} \end{aligned} \tag{3.3}$$

$$\begin{aligned} \boldsymbol{\Phi}' &= \begin{pmatrix} \boldsymbol{\Phi} & \mathbf{0} \\ \mathbf{0} & \varepsilon_{\text{relax}} \end{pmatrix} \quad , \qquad \boldsymbol{\phi}' = \begin{pmatrix} \boldsymbol{\phi} \\ \rho \end{pmatrix} \\ \boldsymbol{\Omega}' &= \begin{pmatrix} \boldsymbol{\Omega} & \mathbf{1} \\ \mathbf{0} & 1 \end{pmatrix} \quad , \qquad \boldsymbol{\omega}' = \begin{pmatrix} \boldsymbol{\omega} \\ 0 \end{pmatrix} \end{aligned} \tag{3.4}$$

$\varepsilon_{\text{relax}}$ は十分に小さい定数とする．

3.3.2 数値計算安定化のための各種パラメータ設定方法

問題で用いる係数行列およびベクトルを次のように設定することで，反復計算の安定性および収束性能を向上させる．

- K_p，K_u，K_{u_z} の各成分を，係るベクトルの各要素をそれぞれ最大値で正規化するように設定する

$$
\begin{aligned}
K_p &= \mathrm{diag}\,\{k_x, k_y, k_\theta, k_l, k_{\theta_{fl}}, k_{\theta_{fr}}\} \\
k_x &= k_y = (\max\{|x_{\mathrm{goal}} - x^0|, |y_{\mathrm{goal}} - y^0|\})^{-2} \\
k_\theta &= (2\pi)^{-2} \\
k_l &= (Y_{\max} - Y_{\min})^{-2} \\
k_{\theta_{fl}} &= k_{\theta_{fr}} = (2\pi)^{-2}
\end{aligned}
\tag{3.5}
$$

$$
\begin{aligned}
K_u &= \mathrm{diag}\,\{\ldots, k_{u_A}, k_{u_B}, k_{u_{l1}}, k_{u_{l2}}, k_{u_{fl}}, k_{u_{fr}}, \ldots\} \\
k_{u_A} &= k_{u_B} = (2\tan^{-1}(X_{\max}/Y_{\min}) + \theta_{2\max})^{-2} \\
k_{u_{l1}} &= k_{u_{l2}} = (Y_{\max} - Y_{\min})^{-2} \\
k_{u_{fl}} &= k_{u_{fr}} = (\theta_{2\max} - \theta_{2\min})^{-2}
\end{aligned}
\tag{3.6}
$$

$$
\begin{aligned}
K_u &= \mathrm{diag}\,\{\ldots, k_{u_{zl}}, k_{u_{zr}}, \ldots\} \\
k_{u_{zl}} &= k_{u_{zr}} = z_{\max}^{-2}
\end{aligned}
\tag{3.7}
$$

- K_u，K_{u_z} は，前項の設定に加えて Sugihara の手法 [20] を適用し，評価関数式 (2.7) の値を用いて重み付けする（I は単位行列，Δk_u はスカラー定数）

$$
\begin{aligned}
K_u &\leftarrow V_{\mathrm{now}}\, K_u + \Delta k_u\, I = \frac{1}{2}\|\varepsilon_{\mathrm{now}}\|_{K_p}^2\, K_u + \Delta k_u\, I \\
K_{u_z} &\leftarrow V_{\mathrm{now}}\, K_{u_z} + \Delta k_{u_z}\, I = \frac{1}{2}\|\varepsilon_{\mathrm{now}}\|_{K_p}^2\, K_{u_z} + \Delta k_{u_z}\, I
\end{aligned}
\tag{3.8}
$$

- 一つ一つの制約条件 $\omega_i(\boldsymbol{q}^0, \boldsymbol{U}_k) \geq 0$ を $\|\nabla\omega_i\|$ で正規化することにより，GI 法内部での反復計算を安定化させ，また緩和問題での各制約条件の比重を揃える [21]
- 緩和問題を解く際，評価関数全体を $\|\boldsymbol{\phi}\|$ で正規化し，元の評価関数部分の値とスラック変数の値の比重を揃える（$\rho = 1$ とできる）

そして，得られた解を用いて Armijo の基準に基づく直線探索を行い，入力列を更新する．

3.3.3 直線探索

直線探索では，次式を満たすステップ幅 α を求める．

$$
\begin{aligned}
H\left(\boldsymbol{q}^0, \boldsymbol{U}_k + \alpha\boldsymbol{\Delta U}_k\right) &\leq H\left(\boldsymbol{q}^0, \boldsymbol{U}_k\right) + \beta\alpha\nabla V\left(\boldsymbol{q}^0, \boldsymbol{U}_k\right)^T \boldsymbol{\Delta U}_k \\
&= H\left(\boldsymbol{q}^0, \boldsymbol{U}_k\right) + \beta\alpha\boldsymbol{\phi}^T \boldsymbol{\Delta U}_k
\end{aligned}
\tag{3.9}
$$

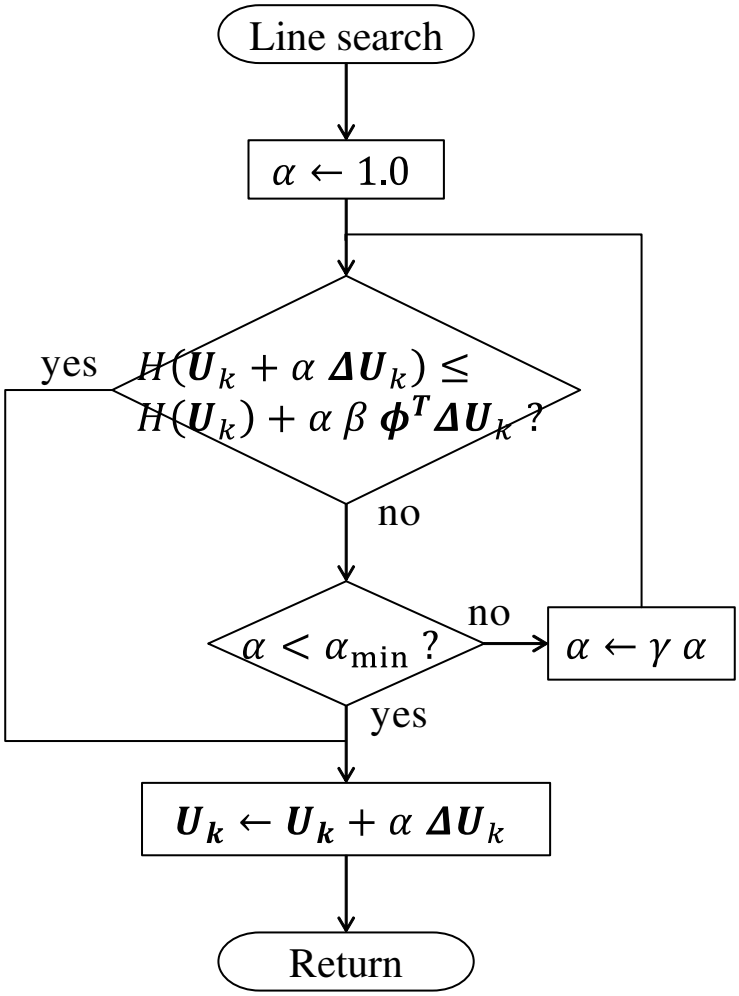


Fig. 3.2　Line search

β は，$0 < \beta < 1$ を満たす定数である．今回，α は Fig. 3.2 に示す単純な探索により求める．なお，γ は $0 < \gamma < 1$ を満たす減衰係数である．

3.4　終了判定

更新後，終了判定を行う．次の項目をすべて満たすことを終了条件とする．

- ε が収束（各要素が許容誤差 $\varepsilon_{\text{goal}}$ の各要素よりも小さい）
- $\boldsymbol{\Delta U}_k$ が収束（各要素が許容誤差 $\varepsilon_{\text{input}}$ の各要素よりも小さい）
- $\omega_i(\boldsymbol{q}^0, \boldsymbol{U}_k) \geq 0$ $(i = 0, \ldots, 20k - 1)$ をすべて満たす

これらの条件を全て満たしていれば，その時点での $\boldsymbol{U}_k$ を解とし，計画を終了する．満たし

ていない場合は次節の処理へ移る．

3.5 目標到達可否判定

初期ステップ数 k_{init} は直線距離から算出するため，現在の k では歩数不足の場合がある．そこで，反復毎に目標状態へ収束する見込みがあるかどうかを判定し，見込みがない場合に k を増やす処理を行う．ここでは，次の項目のいずれかを満たせば到達不可と判定することとする．

- “$\boldsymbol{\Delta U}_k$ 収束時に ε が未収束” という状況が $n_{\text{reachable1}}$ 回繰り返される
- 現在の k における反復回数 loop が $n_{\text{reachable2}}$ に達した段階で，ε が大きい値をとる（各要素が到達見込みの閾値 $\varepsilon_{\text{reachable}}$ の各要素よりも大きい）

到達不可と判定された場合には $k \leftarrow k+1$ とし，また $\boldsymbol{U}_k$ はその内容を引き継ぎ $\boldsymbol{U}_k \leftarrow (\boldsymbol{U}_k^T, \boldsymbol{0}^T)^T$ として再度反復を開始する．そうでなければ，“Solve QPP” へ戻り反復を継続する．

3.6 地形情報の構築

前述の計画アルゴリズムを開始する前に，地形適応制約で用いる地形情報 $z_{\text{field}}(x, y)$ を構築する必要がある．深度カメラや測域センサから得られる点群データを用いて地形情報を構築することを想定し，次の手順で点群データからの構築を行った．

1. xy 平面を等間隔に幅 D_{grid} 四方の正方形グリッドで分割する．
2. 各グリッドの領域に含まれるサンプルデータ群の z 軸成分の平均値を，グリッド中央における高さとする．
3. x, y 方向それぞれについて，グリッド中心の点列をサンプルとする 3 次スプライン補間関数を計算．
4. $z_{\text{field}}(x, y)$, $\partial z_{\text{field}}/\partial x$, $\partial z_{\text{field}}/\partial y$ は，与えられた点 (x, y) に対して直近の x 軸方向 2 本，y 軸方向 2 本の補間関数上の値の距離重み付け平均で算出する．

なお，シミュレーションにおいて関数からサンプリングを行って点群データを得る場合，D_{sample} 間隔でサンプリングを行う．

3.7　本章のまとめ

本章では，二次計画法を用いた反復法による脚配置計画問題の求解アルゴリズムの要素として，必要歩数の推定から毎回の二次計画問題の求解と緩和問題の設定，収束判定，収束見込み判定とステップ数追加について説明した．また，地形の違いなどにより問題ごとに大きく変化してしまうことを可能な限り回避するため，反復法における数値計算の性能を安定させる重み行列の設定方法などについても述べた．

次章以降は，提案する脚配置計画手法の有用性および性能を検証するための計画シミュレーションおよび実機実験について述べていく．

第 4 章

HRP-2 における
脚配置計画シミュレーション

本章および次章では，提案手法による脚配置計画によって，三次元地形上でどのような計画が得られるのか，また実時間で計画が行えるかどうかを検証する．本章および次章で示す結果は，CPU: Intel® Core™ i7-2640M2.8 GHz × 2，RAM:8 GB の PC を用い，Visual Studio 2013 上で Visual C++ で作成したプログラムを実行して得られたものである．計算時間は 100 回計画を行ったときの平均値を示している．

本章では，既存研究 [17] でも用いられていたヒューマノイドロボット HRP-2 [23] を想定した計画シミュレーションとその結果について記述する．地形形状が計画に与える影響を検証するため，滑らかに連続的に変化する形状をもつ地形と，不連続的に高さの変化する形状をもつ地形とでそれぞれ計画を行い，結果について考察する．

4.1 各種パラメータの設定

手法 [17] に合わせて各種パラメータを設定する．$z_{\max}$ については，歩行の安定性を考慮し 10.0 cm とした．パラメータは Table 4.1 および Table 4.2 のとおりである．

4.2 連続的変化のある地形上での計画

連続的変化のある地形として山型のものとスロープ型のものとを考えることとし，それらの地形上での計画結果を確認する．なお，地形構築の際のグリッドマップのサイズと関数からのサンプリング間隔は Table 4.3 に示す通りそれぞれ 5.0 cm，1.0 cm とする．

Table 4.1 Model parameters for HRP-2

parameters	value
$z_{\max}$	10.0 cm
$d_{xy\,\max}$	50.0 cm
$d_{z\,\max}$	20.0 cm
walking period (whole)	900.0 ms
walking period (single support)	800.0 ms
walking period (double support)	100.0 ms
$\theta_{2\max}$	15°
$\theta_{2\min}$	−45°
$X_{\max}$	23.38 cm
$X_{\min}$	−23.38 cm
$Y_{\max}$	27.0 cm
$Y_{\min}$	13.5 cm
O_{xf}	13.39 cm
O_{xb}	10.75 cm
O_{yi}	5.9 cm
O_{yo}	7.9 cm
a	-4.50926×10^{-12}
b	1.32
c	7.08

Table 4.2 Algorithm parameters for HRP-2

parameters	value
Δk_u	10^{-5}
Δk_{u_z}	10^{-5}
$\varepsilon_{\text{relax}}$	10^{-15}
$\alpha_{\min}$	0.15
β	0.5
γ	0.95
$\varepsilon_{\text{goal}}$	$(0.1, 0.1, 0.1, 0.1, 0.1, 0.1)^T$ (cm, deg)
$\varepsilon_{\text{input}}$	$(\ldots, 10^{-6}, 10^{-6}, 10^{-6}, 10^{-6}, 10^{-6}, 10^{-6}, \ldots)^T$ (cm, deg)
$\varepsilon_{\text{reachable}}$	$(2.5, 2.5, 0.5, 1.0, 0.5, 0.5)^T$ (cm, deg)
$n_{\text{reachable1}}$	10
$n_{\text{reachable2}}$	$3k$

Table 4.3　Map parameters for HRP-2 (continuous field)

parameters	value
D_{grid}	5.0 cm
D_{sample}	1.0 cm

Table 4.4　Result of planning (single hill)

z_{top} (cm)	k_{init}	k	iteration	time(ms)
plane	10	10	19	11.67
25.0	10	10	31	21.06
100.0	10	11	53	40.67

4.2.1　単一の山を配置した場合

初期状態 $\boldsymbol{q}^0 = (0.0, 0.0, 0.0, 13.5, 0.0, 0.0)^T$ (cm, deg) から目標状態 $\boldsymbol{q}_{\text{goal}} = (300.0, 300.0, 0.0, 13.5, 0.0, 0.0)^T$ へ向かう計画を行う．山型の地形は，次の関数で定義する．

$$z_{\text{field}}(x,\, y) = z_{\text{top}} \exp\left(-\frac{(x - x_c)^2 + (y - y_c)^2}{2.0\,(30.0)^2}\right) \tag{4.1}$$

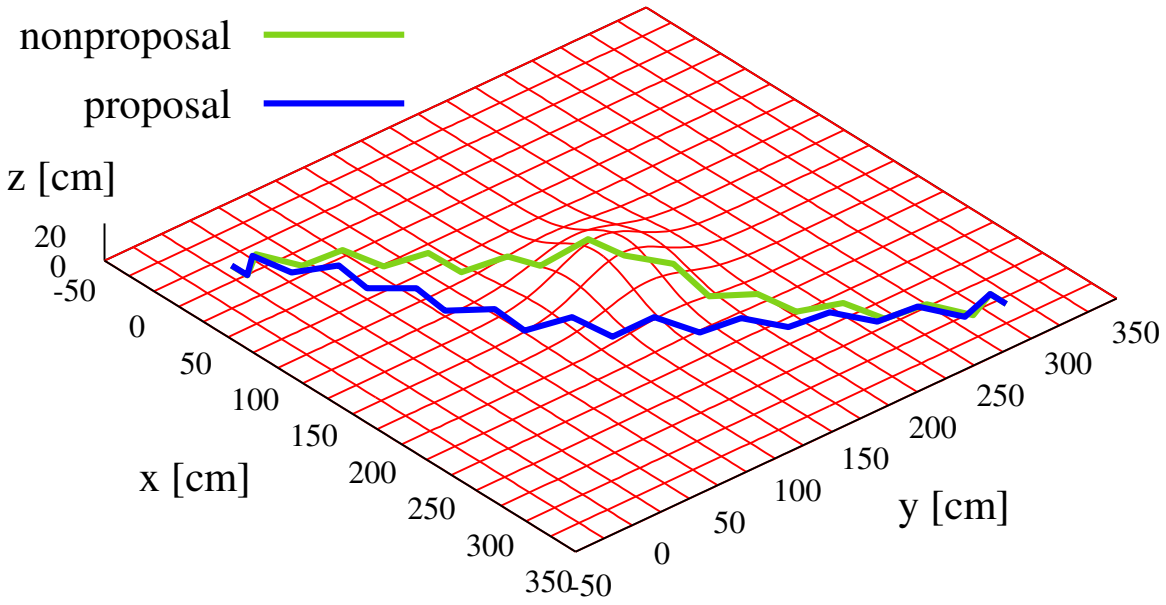
x_c, y_c は山の中心，z_{top} は頂点の高さである．$x_c = 150.0$ cm, $y_c = 150.0$ cm に一つ山を配置し，$z_{\text{top}} = 25.0$, 100.0 cm の 2 パターンについて計画を行った．

Figure 4.1，Fig. 4.2，Fig. 4.3，Table 4.4 に計画結果を示す．$z_{\text{top}} = 25.0$ cm の場合，初期ステップ数 $k_{\text{init}} = 10$ のまま，地形適応制約を守るために迂回するように歩行して目標状態へ到達する計画が得られている．$z_{\text{top}} = 100.0$ cm の場合，初期ステップでは迂回するためには歩数が不足しているが，収束見込み判定が機能し，ステップ数を増加させることで目標へ到達する解を得られている．これに伴い，反復回数と計算時間は若干増加している．

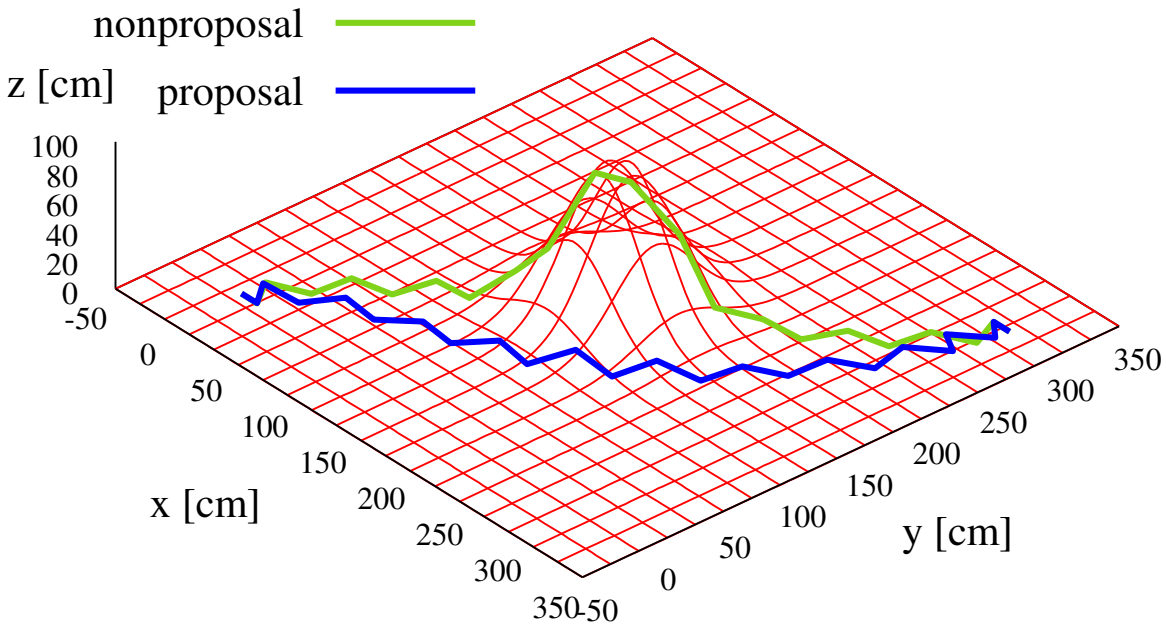
4.2.2　二つの山を配置した場合

次に，二つの山を配置した場合のシミュレーションを行った．

$$\begin{aligned} z_{\text{field}}(x,\, y) &= z_{\text{top}} \exp\left(-\frac{(x - x_{c1})^2 + (y - y_{c1})^2}{2.0\,(30.0)^2}\right) \\ &= z_{\text{top}} \exp\left(-\frac{(x - x_{c2})^2 + (y - y_{c2})^2}{2.0\,(30.0)^2}\right) \end{aligned} \tag{4.2}$$

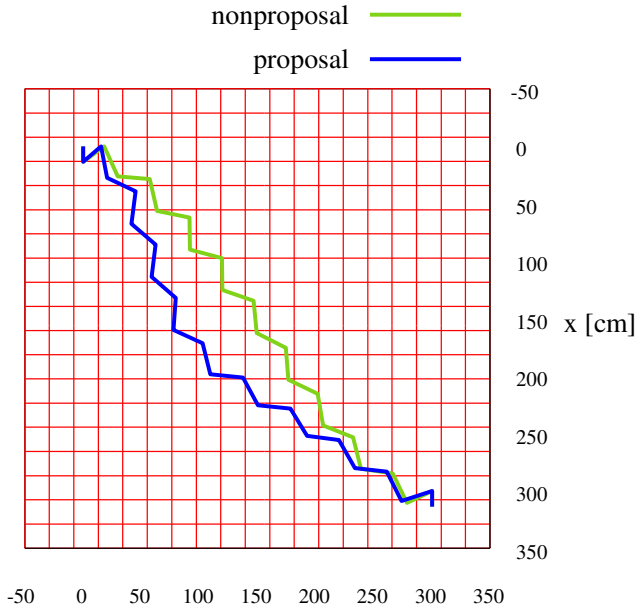


(a) $z_{\text{top}} = 25.0\,\text{cm}$

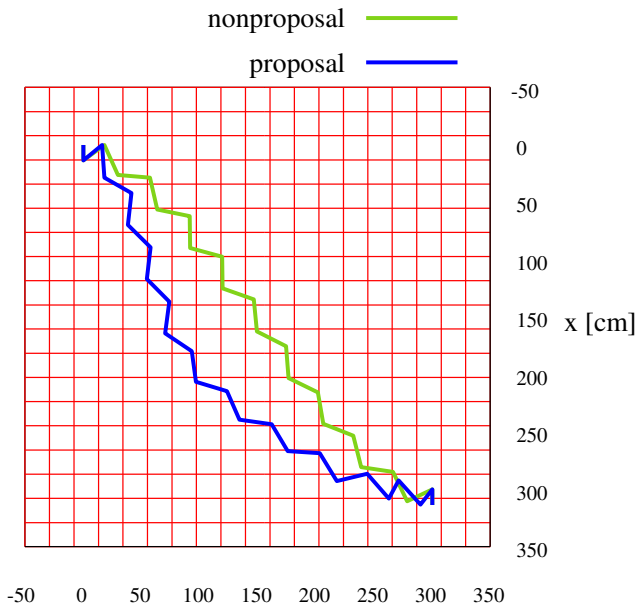


(b) $z_{\text{top}} = 100.0\,\text{cm}$

Fig. 4.1 Footsteps$x$-y-$z$view (single hill)

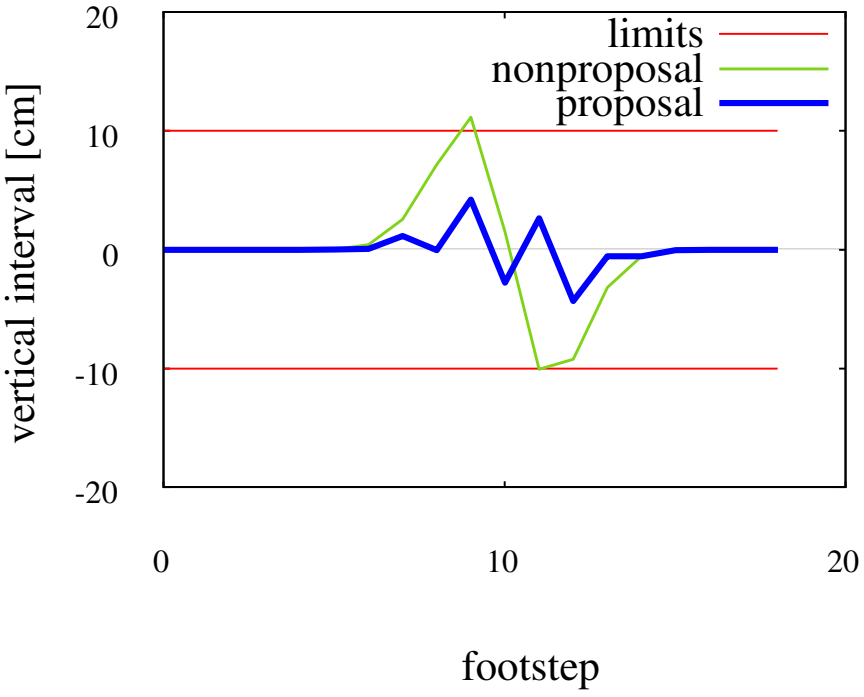


(a) $z_{\text{top}} = 25.0\,\text{cm}$

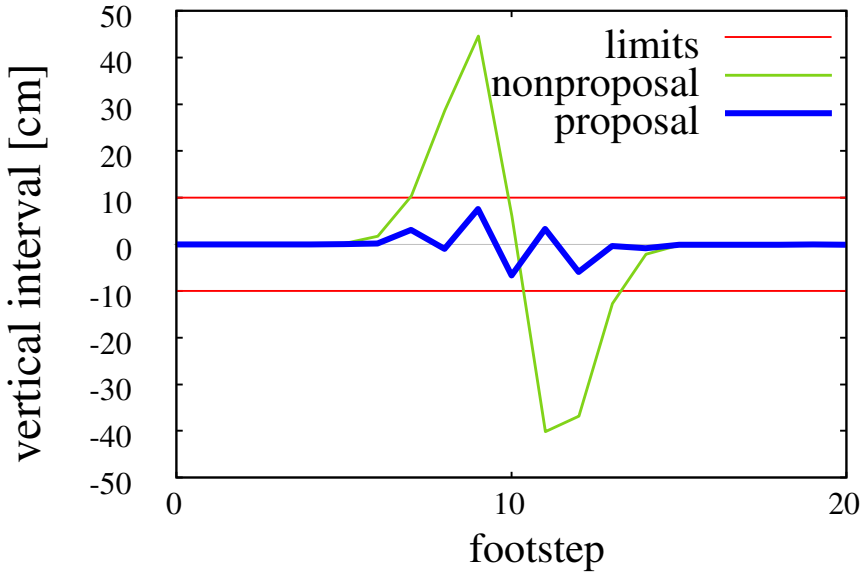


(b) $z_{\text{top}} = 100.0\,\text{cm}$

Fig. 4.2 Footsteps*x*-*y*view (single hill)



(a) $z_{\text{top}} = 25.0\,\text{cm}$



(b) $z_{\text{top}} = 100.0\,\text{cm}$

Fig. 4.3 Vertical interval (single hill)

Table 4.5　Result of planning (double hill)

$(x_c,\ y_c)$ (cm)	k_{init}	k	iteration	time(ms)
plane	10	10	19	11.67
(75.0, 150.0), (225.0, 0.0)(cm)	10	10	22	14.28
(75.0, 300.0), (225.0, 150.0)(cm)	10	10	22	14.19

Table 4.6　Result of planning (slope)

θ_{slope} (deg)	k_{init}	k	iteration	time(ms)
plane	11	11	14	7.45
5.0	11	11	20	14.11

二つとも $z_{\text{top}} = 100.0\,\text{cm}$ とし，山の配置 $(x_{c1},\ y_{c1})$，$(x_{c2},\ y_{c2})$ を (75.0, 150.0), (225.0, 0.0)(cm) とした場合と，(75.0, 300.0)，(225.0, 150.0)(cm) とした場合の 3 パターンで計画を行った．

Figure 4.4，Fig. 4.5，Fig. 4.6，Table 4.5 に計画結果を示す．Figure 4.4a，Fig. 4.4b いずれにおいても山の間を通って最短で目標へ到達する脚配置が計画されており，Figure 4.6a, Fig. 4.6b を見ても脚先の垂直移動は制約の範囲内に収まっていることから，地形に合わせて最適な経路をとる脚配置が計画できていることがわかる．

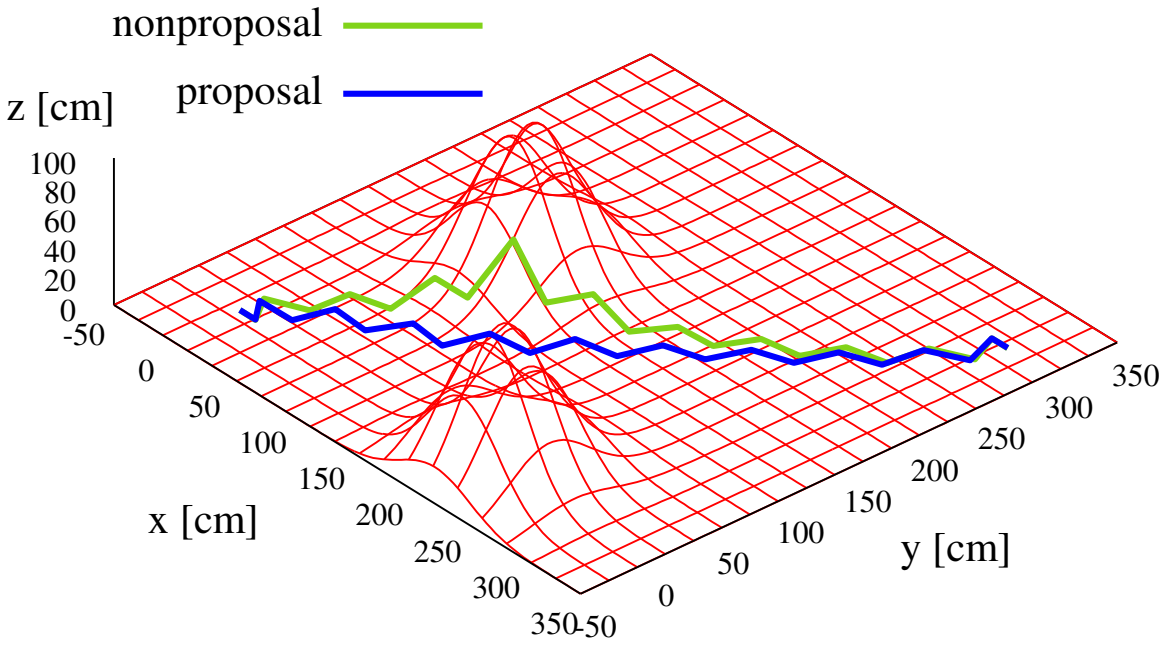
4.2.3　スロープ型の地形でのシミュレーション

初期状態 $\boldsymbol{q}^0 = (0.0,\ 0.0,\ 0.0,\ 13.5,\ 0.0,\ 0.0)^T$ (cm, deg) から目標状態 $\boldsymbol{q}_{\text{goal}} = (150.0,\ 450.0,\ 0.0,\ 13.5,\ 0.0,\ 0.0)^T$ へと向かう計画を行う．地形としては，y 軸方向において平面と傾斜角 θ_{slope} の斜面が $y = 100\,\text{cm}$ の位置で接続しているスロープ型の地形を用いる．スロープは $y = 100\,\text{cm}$ の位置で終了し，平面形状へと戻る．

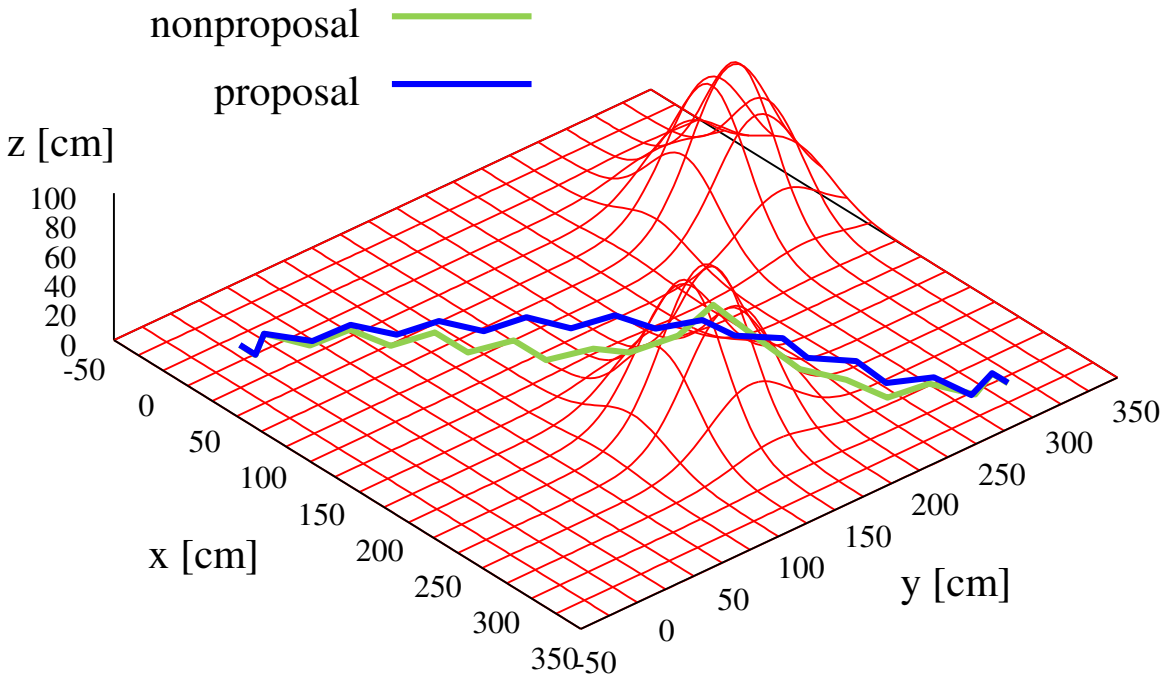
$$z_{\text{field}}(x,\ y) = \begin{cases} 0.0 & (y < 100.0) \\ (y - 100.0)\tan\theta_{\text{slope}} & (100.0 \le y < 350.0) \\ 250.0\tan\theta_{\text{slope}} & (350.0 \le y) \end{cases} \tag{4.3}$$

$\theta = 5.0°$ で計画を行う．

Figure 4.7，Fig. 4.8，Fig. 4.9，Table 4.6 に計画結果を示す．地形を考慮しない場合でも地形適応制約を破らない範囲であるため，脚配置や計算時間，反復回数などに大きな差はない．提案手法においては反復計算の過程で垂直移動の仮想的な入力の最小化評価が行われて

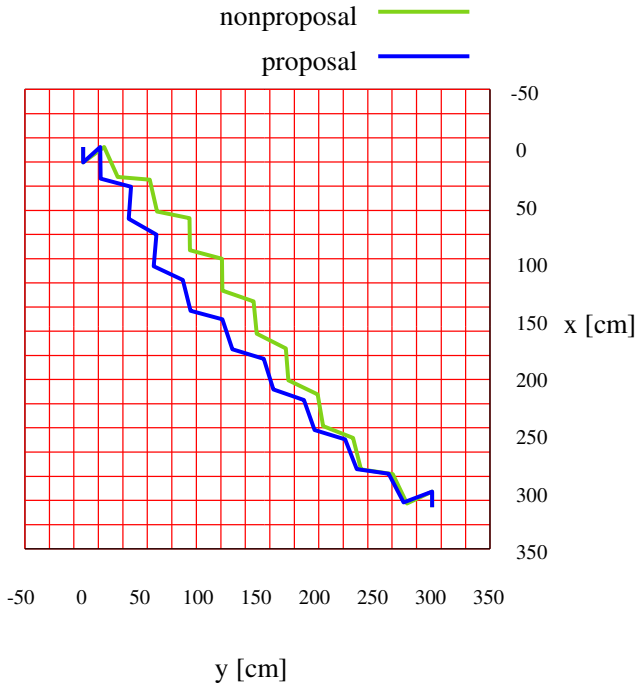


(a) (75.0, 150.0), (225.0, 0.0)(cm)

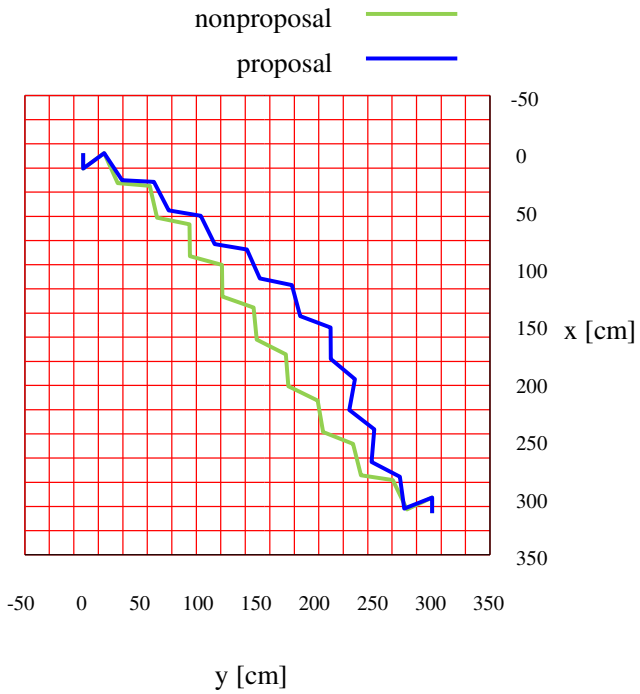


(b) (75.0, 300.0), (225.0, 150.0)(cm)

Fig. 4.4 Footsteps*x*-*y*-*z*view (double hill)

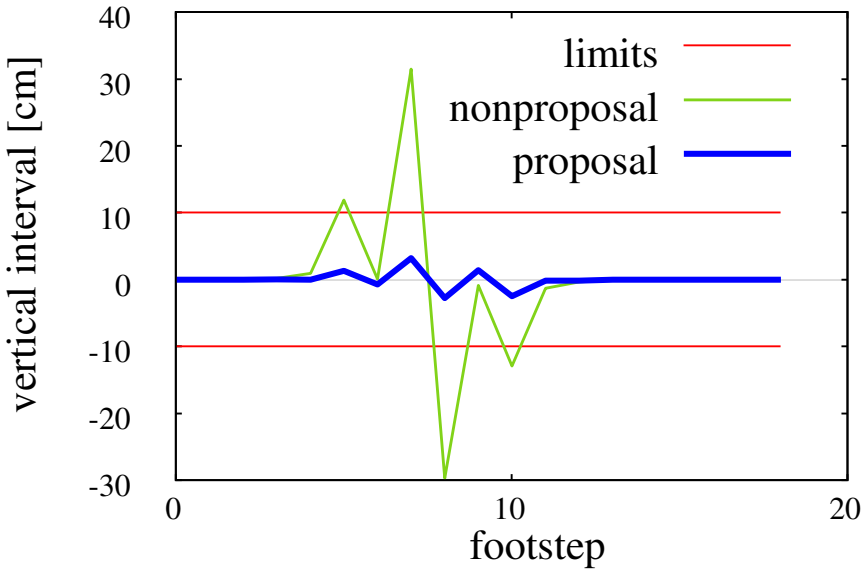


(a) (75.0, 150.0), (225.0, 0.0)(cm)

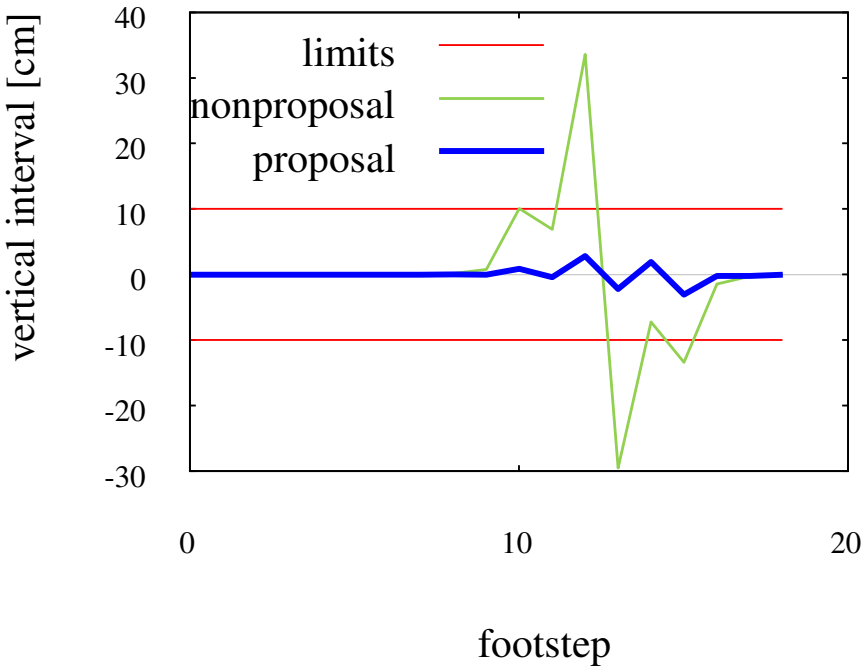


(b) (75.0, 300.0), (225.0, 150.0)(cm)

Fig. 4.5 Footsteps*x*-*y*view (double hill)



(a) (75.0, 150.0), (225.0, 0.0)(cm)



(b) (75.0, 300.0), (225.0, 150.0)(cm)

Fig. 4.6 Vertical interval (double hill)

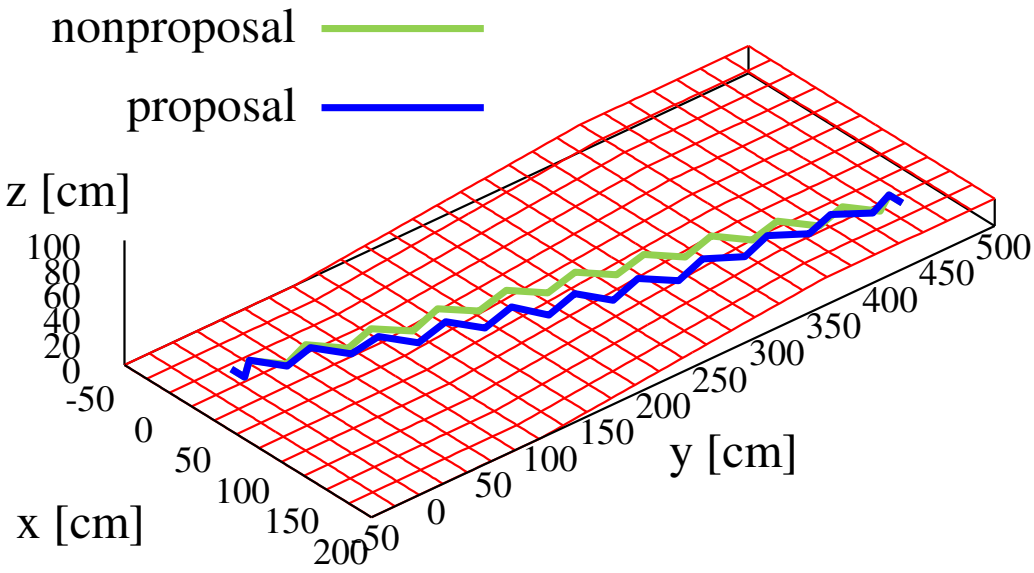


Fig. 4.7　Footsteps x-y-z view (slope)

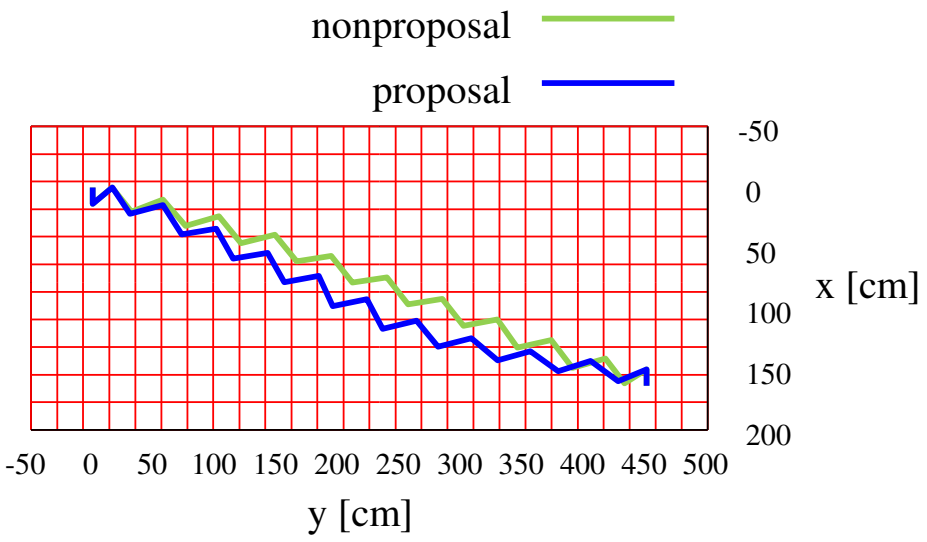


Fig. 4.8　Footsteps x-y view (slope)

いるため，平面上での計画と比較すると若干経路が変化している．この結果から，水平方向の動作に加えて垂直方向の動作も正しく評価され，最適化されていることがわかる．

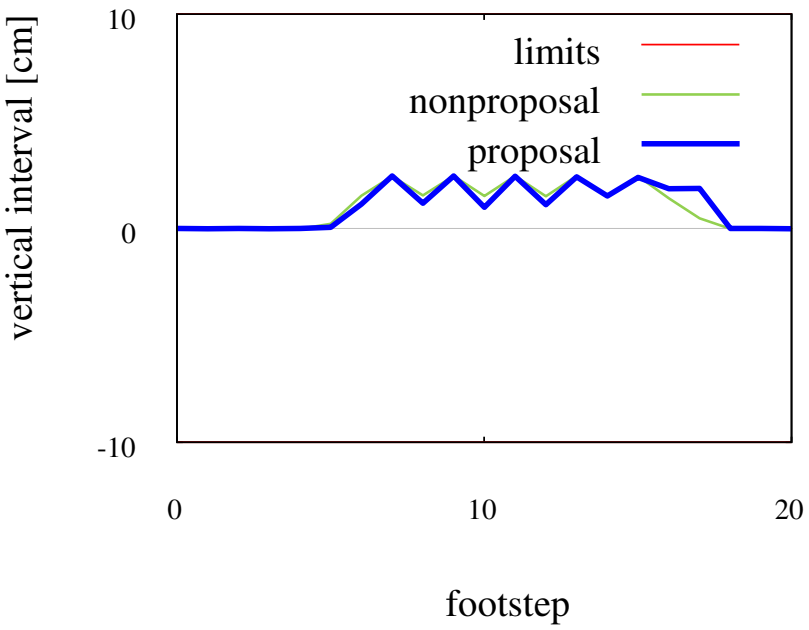


Fig. 4.9　Vertical interval (slope)

Table 4.7　Map parameters for HRP-2 (discontinuous field)

parameters	value
D_{grid}	1.0 cm
D_{sample}	0.2 cm

4.2.4　計算時間の評価

適用する二足歩行ロボットの一歩進むのにかかる時間よりも計算時間が十分に短ければ，オンラインでの再計画が可能となり，実時間性がある．今回シミュレーションした地形においては，いずれの結果においても実機 HRP-2 の歩行周期全体のみならず，両脚支持期間よりも短い時間で計画できており，連続的変化のある地形においては実時間性があることが示された．

4.3　不連続な変化のある地形上での計画

不連続な変化のある地形として，段差のある地形と階段状の地形を考えることとし，それらの地形上での計画結果を確認する．なお，地形構築の際のグリッドマップのサイズと関

数からのサンプリング間隔は，不連続な変化がある部分に対応するセルが小さくなるよう，Table 4.7 に示す通りそれぞれ 1.0 cm，0.2 cm とする．段差や階段のような変化が含まれるセルでは，領域内のサンプル点の平均をとったときに実際の地形には存在しない，段と段との間の中途半端な値が表現されてしまうことになる．現在の地形適応の方法および地形情報構築方法ではこれは避けられないため，できるだけグリッドサイズを小さくすることでこれに対応することとする．

4.3.1 上り階段地形

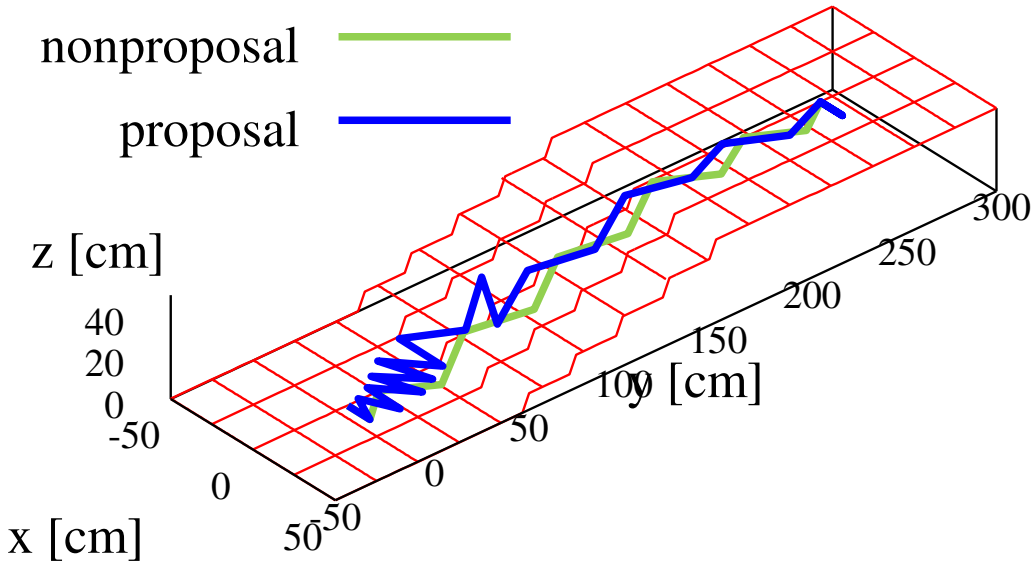
初期状態 $\boldsymbol{q}^0 = (0.0, 0.0, 0.0, 13.5, 0.0, 0.0)^T$ (cm, deg) から目標状態 $\boldsymbol{q}_{\text{goal}} = (0.0, 250.0, 0.0, 13.5, 0.0, 0.0)^T$ へと向かう計画を行う．地形としては，$yy = 50.0$ cm から始まり y 方向へ 25.0 cm 進むごとに 8.0 cm ずつ上る 5 段の上り階段とする．

$$z_{\text{field}}(x,\ y) = \begin{cases} 0.0 & (y < 50.0) \\ 8.0 & (50.0 \leq y < 75.0) \\ 16.0 & (45.0 \leq y < 100.0) \\ 24.0 & (100.0 \leq y < 125.0) \\ 32.0 & (125.0 \leq y < 150.0) \\ 40.0 & (150.0 \leq y) \end{cases} \tag{4.4}$$

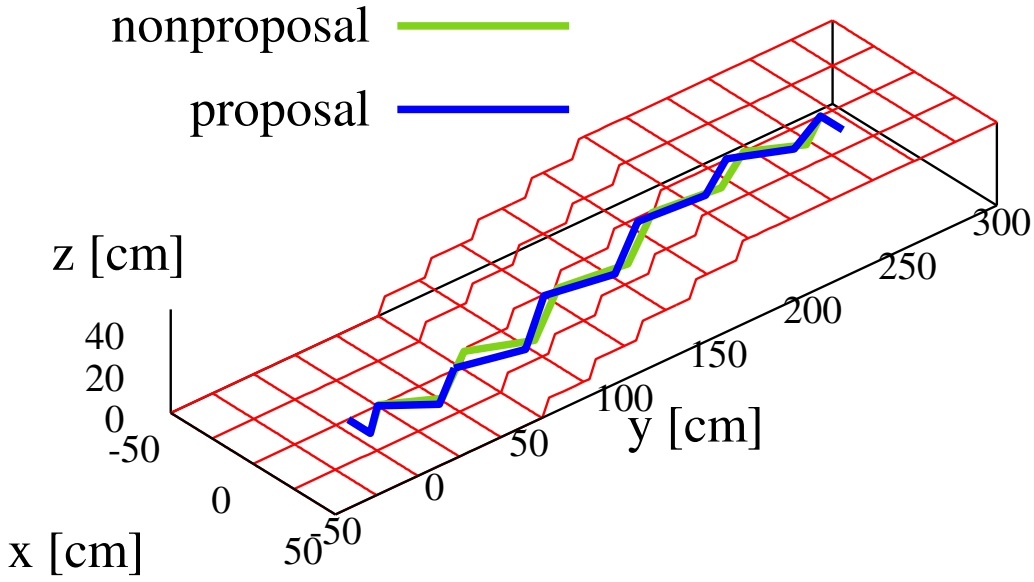
また，それを半歩 $d_{xyxy}/4 = 12.5$ cm だけ前方にずらした地形でも計画を行う．

$$z_{\text{field}}(x,\ y) = \begin{cases} 0.0 & (y < 62.5) \\ 8.0 & (62.5 \leq y < 87.5) \\ 16.0 & (87.5 \leq y < 112.5) \\ 24.0 & (112.5 \leq y < 137.5) \\ 32.0 & (137.5 \leq y < 162.5) \\ 40.0 & (162.5 \leq y) \end{cases} \tag{4.5}$$

Figure 4.10，Fig. 4.11，Fig. 4.12，Table 4.8 に計画結果を示す．2 パターンの配置のうち，前者では初期位置付近で余計に歩行し，同じ段に二歩の間留まるといった挙動を含む計画となっている．これは，現在の解法は不連続な制約条件の変化に弱く求解性能が落ちるため，反復計算の初期で局所解に陥っているためである．しかし，わずかに地形の変化した後者においては初期ステップ数 6 のまま 14 回の反復で解に収束していること，またいずれの計画



(a) Reference placement

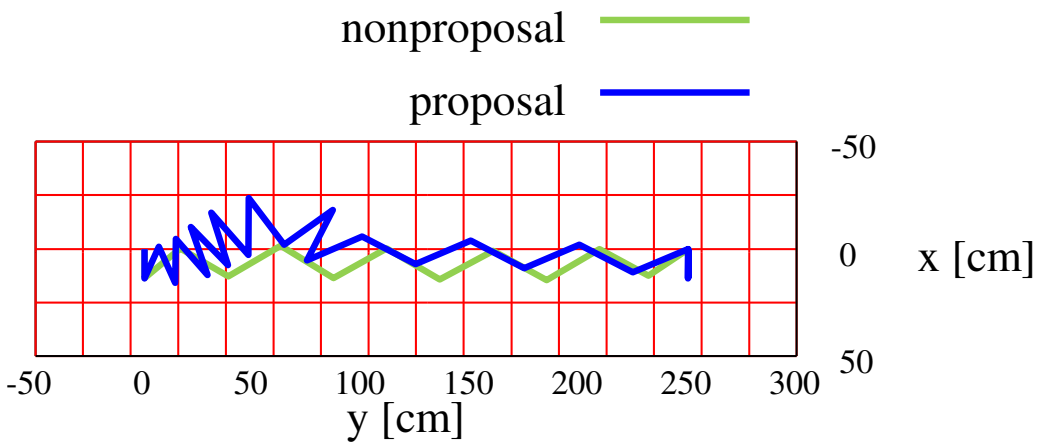


(b) Placement to a half step length forward

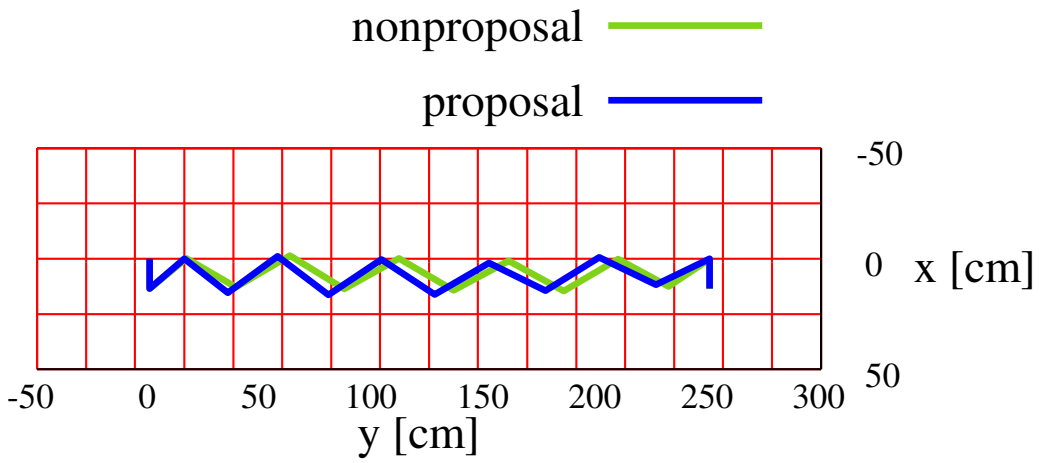
Fig. 4.10　Footsteps*x*-*y*-*z*view (upward stairway)

も両脚支持期間以下の時間で計算を終えられていることから，最適化計算手法の修正や変更で改善できると考えられる．

また，Fig. 4.12a を見ると，両脚間の高低差は 8.0 であることが理想であるが異なる値をとっていることがわかる．これは，グリッドマップ上で不連続な変化をしている段差付近の



(a) Reference placement

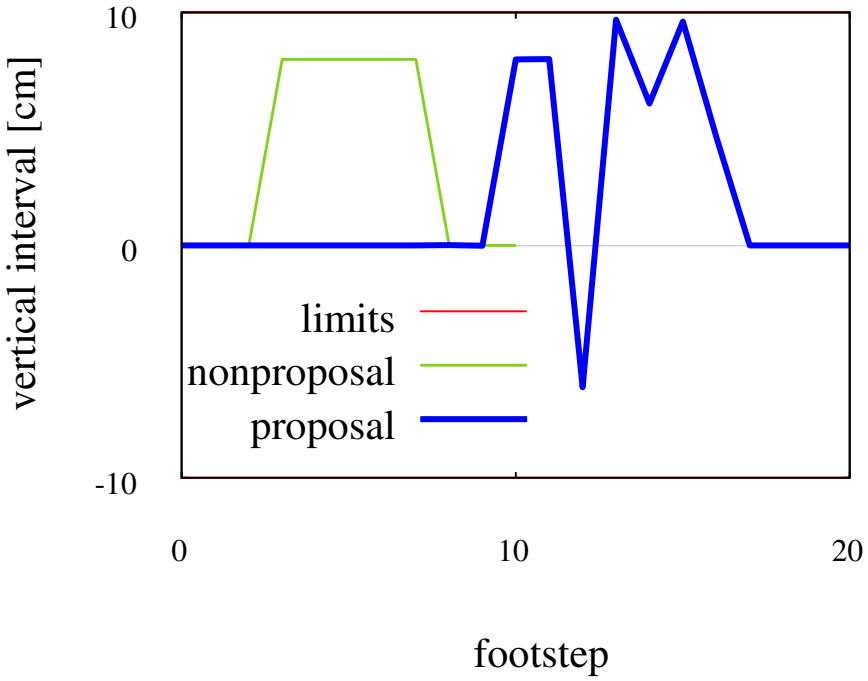


(b) Placement to a half step length forward

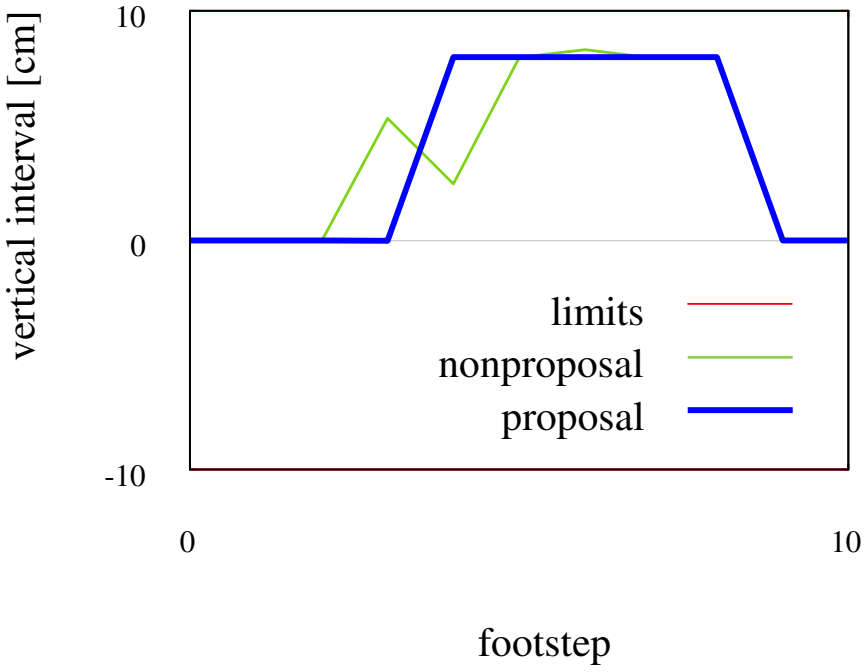
Fig. 4.11 Footsteps$x$-$y$view (upward stairway)

Table 4.8 Result of planning (upward stairway)

	k_{init}	k	iteration	time(ms)
plane	6	6	12	1.85
reference	6	11	164	59.63
half step forward	6	6	14	2.11



(a) Reference placement



(b) Placement to a half step length forward

Fig. 4.12 Vertical interval (upward stairway)

セルを踏み，地形形状を表現する補間関数が本来の形状にない値を返してしまうためである．この問題に対しては，脚先の三次元姿勢をモデルとして考慮し，これに対して制限制約をかけることで解決できると考えられる．三次元姿勢を考慮することで，不連続な変化のある地形付近の，補間関数の傾きが急峻となる領域を回避するような計画となると考えられるためである．

4.3.2 下り階段地形

初期状態 $\boldsymbol{q}^0 = (0.0, 0.0, 0.0, 13.5, 0.0, 0.0)^T$ (cm, deg) から目標状態 $\boldsymbol{q}_{\text{goal}} = (0.0, 250.0, 0.0, 13.5, 0.0, 0.0)^T$ へと向かう計画を行う．地形としては，$yy = 50.0\,\text{cm}$ から始まり y 方向へ 25.0 cm 進むごとに 8.0 cm ずつ下る 5 段の下り階段とする．

$$z_{\text{field}}(x,\ y) = \begin{cases} 40.0 & (y < 50.0) \\ 32.0 & (50.0 \leq y < 75.0) \\ 24.0 & (75.0 \leq y < 100.0) \\ 16.0 & (100.0 \leq y < 125.0) \\ 8.0 & (125.0 \leq y < 150.0) \\ 0.0 & (150.0 \leq y) \end{cases} \tag{4.6}$$

また，それを半歩 $d_{xyxy}/4 = 12.5\,\text{cm}$ だけ前方にずらした地形でも計画を行う．

$$z_{\text{field}}(x,\ y) = \begin{cases} 40.0 & (y < 62.5) \\ 32.0 & (62.5 \leq y < 87.5) \\ 24.0 & (87.5 \leq y < 112.5) \\ 16.0 & (112.5 \leq y < 137.5) \\ 8.0 & (137.5 \leq y < 162.5) \\ 0.0 & (162.5 \leq y) \end{cases} \tag{4.7}$$

Figure 4.13，Fig. 4.14，Fig. 4.15，Table 4.9 に計画結果を示す．上り階段地形と同様に，半歩分の配置の違いで結果に差が出ており，不連続な変化の地形に対しては計算が不安定となり求解性能が低下していることがわかる．

Table 4.9　Result of planning (downward stairway)

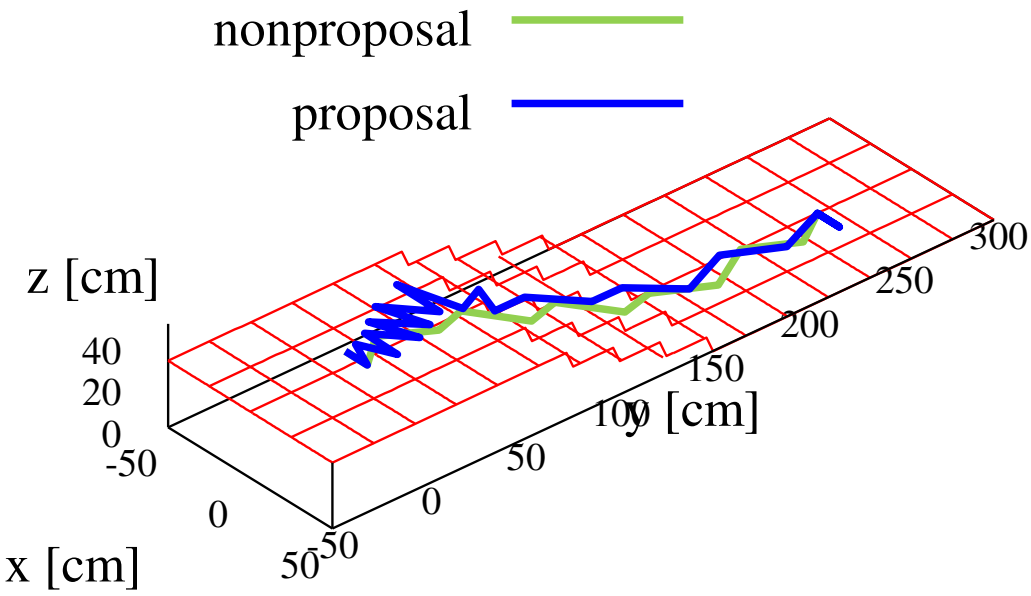
	k_{init}	k	iteration	time(ms)
plane	6	6	12	1.82
reference	6	11	164	62.91
half step forward	6	6	14	2.00

4.3.3　計算時間についての評価

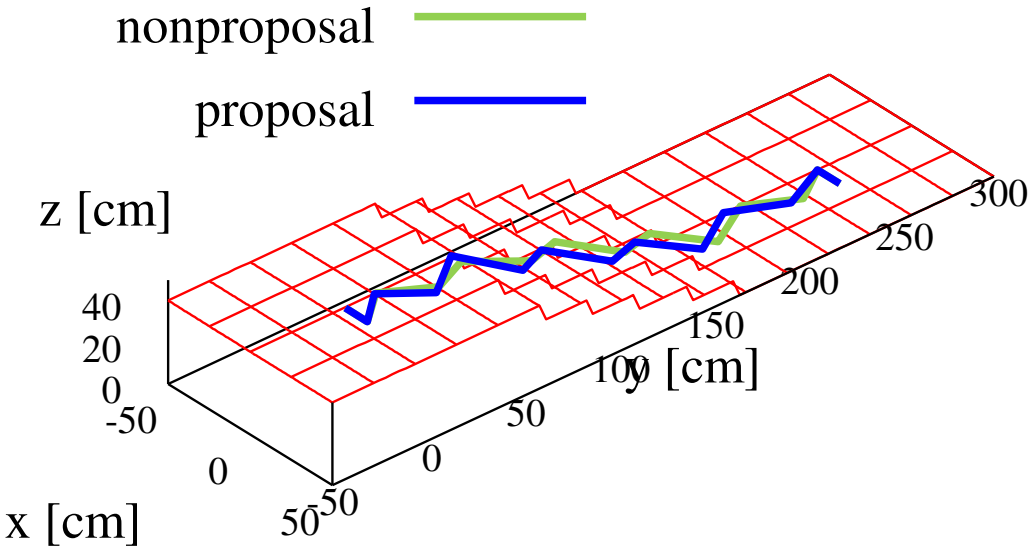
不連続な地形に対しては計算が不安定化してしまうことが明らかとなったが，いずれの結果においても計算自体は両足支持期間よりも短く，実時間性は保たれている．これは，計画アルゴリズムにおける収束見込み判定とステップ数の増補により，問題が再定義されたことで局所解から脱することができたためである．

4.4　本章のまとめ

本章では，HRP-2 に基づいたパラメータを用いて，連続的変化のある地形と不連続な地形とでいくつかの問題設定で計画を行った．そして，不連続な地形上では計算が不安定化してしまう問題が残るが，連続的な地形では安定して最適化された脚配置計画を得ることができ，またいずれにおいても計算時間は両脚支持期間よりも短く，実時間性を有していることが示された．

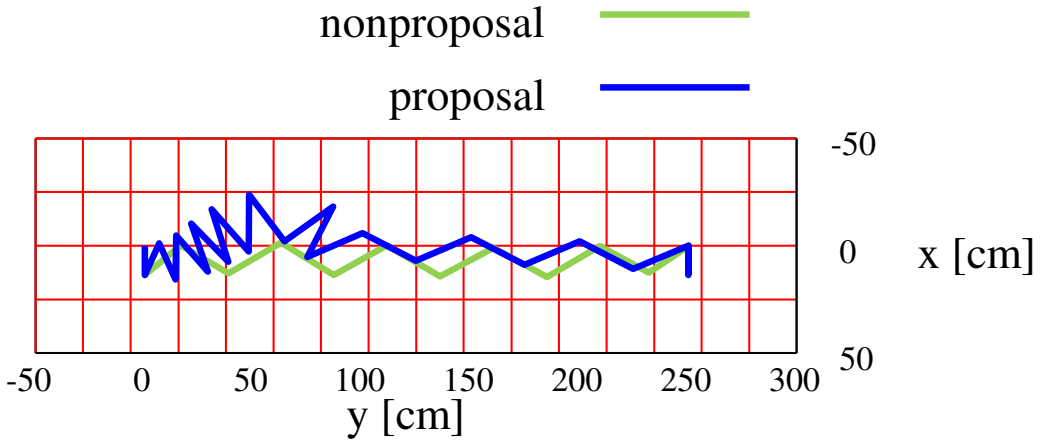


(a) Reference placement

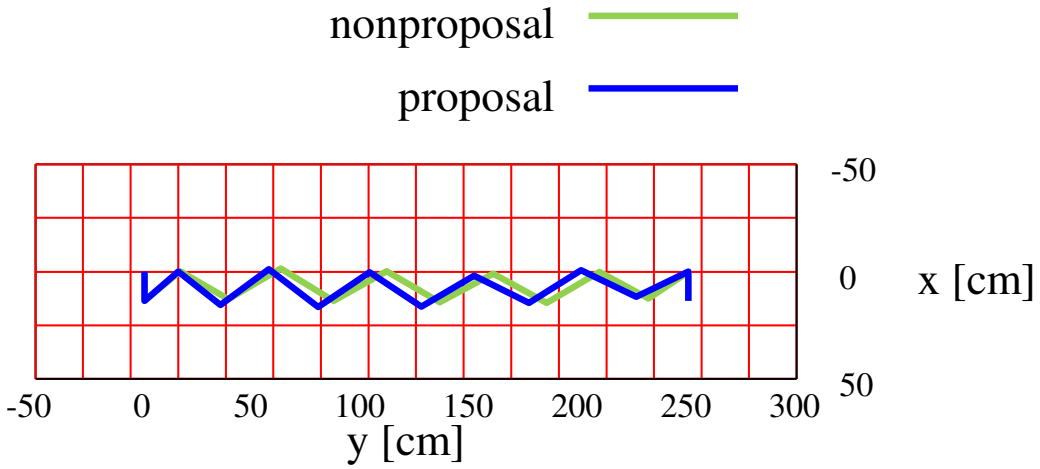


(b) Placement to a half step length forward

Fig. 4.13 Footsteps*x*-*y*-*z*view (downward stairway)

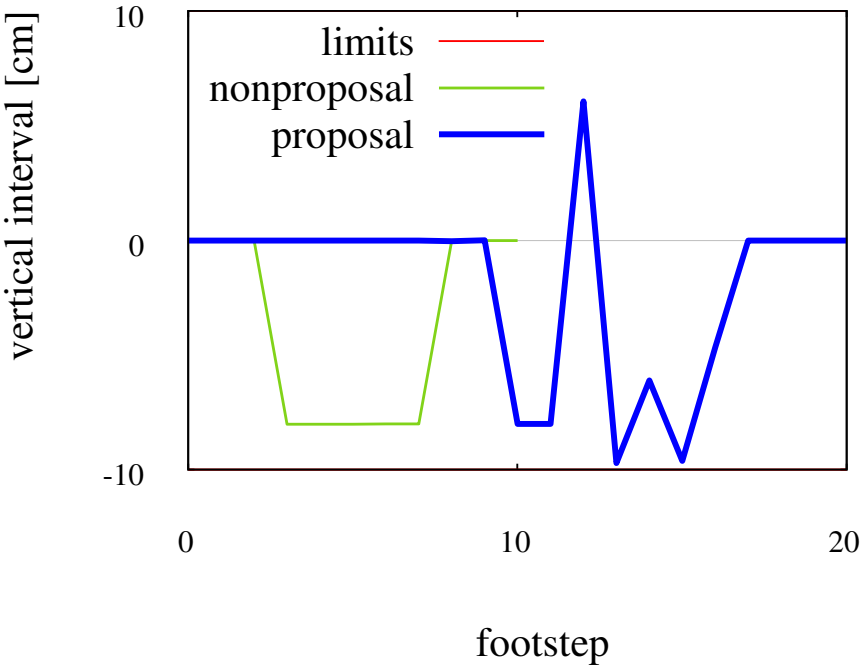


(a) Reference placement

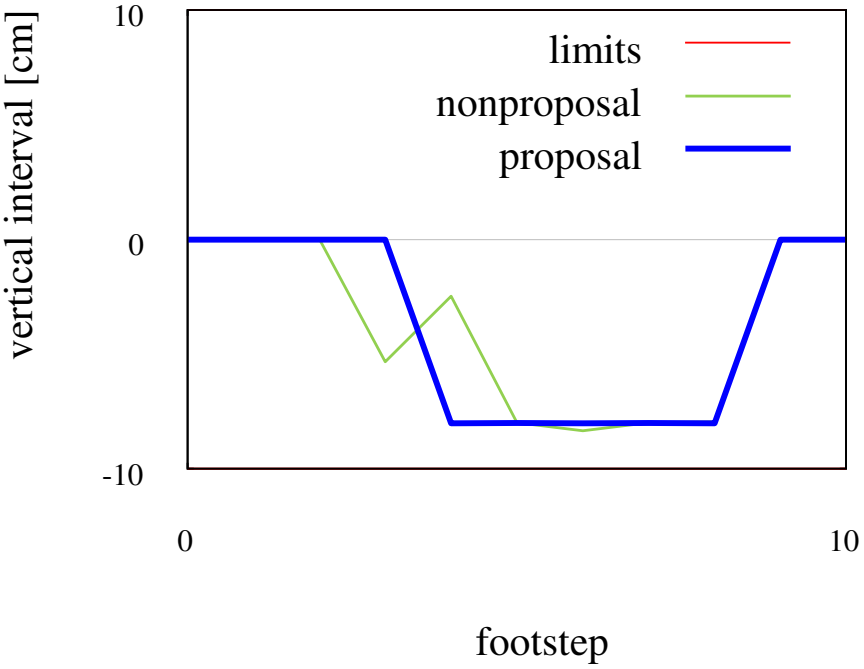


(b) Placement to a half step length forward

Fig. 4.14 Footsteps$x$-$y$view (downward stairway)



(a) Reference placement



(b) Placement to a half step length forward

Fig. 4.15 Vertical interval (downward stairway)

第 5 章

NAO における
脚配置計画シミュレーション

本章では，Aldebaran Robotics のヒューマノイドロボット NAO [24] を用いた計画シミュレーションおよび実機実験，また kinect v2 [27] で計測した地形情報を用いた計画シミュレーションについて記述する．

5.1 踏破性能とパラメータ設定

パラメータとしては，NAO に標準搭載されているシステム NAOqi が提供する脚配置指定による歩行指示の API を用いることを前提に，それらを利用する際に定められている限界値を各制約条件のパラメータとして設定する [26]．NAO の歩行制御については，標準の制御用 API としては二次元平面上の歩行を前提としたものしか提供されておらず，床面の上下方向の誤差に対するロバスト性を持った二次元平面上における歩行制御が行われている [25]．そのため，本章におけるシミュレーションおよび実験では，提案手法における垂直方向移動の上下限値には，脚先の上下方向の誤差に対するロバスト性において耐えうると [25] で示されている誤差の許容値を設定する．検証内容としては，三次元の地形形状を持つフィールドにおいて，ロバスト性の範囲内で踏破可能となる，目標状態までの最適化された脚配置を計画できることを検証するものとする．

パラメータは Table 5.1，Table 5.2 および Table 5.3 のとおりである．

Table 5.1 Model parameters for NAO

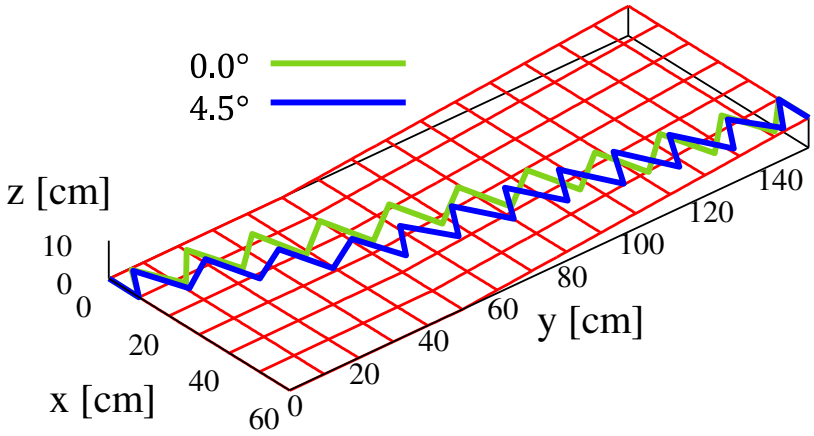
parameters	value
$z_{\max}$	0.5 cm
$d_{xy\,\max}$	16.73 cm
$d_{z\,\max}$	1.0 cm
walking period (whole)	420–600 ms
$\theta_{2\max}$	30°
$\theta_{2\min}$	−30°
$X_{\max}$	6.0 cm
$X_{\min}$	−4.0 cm
$Y_{\max}$	16.0 cm
$Y_{\min}$	8.8 cm
O_{xf}	10.69 cm
O_{xb}	5.61 cm
O_{yi}	3.95 cm
O_{yo}	5.23 cm
a	-1.18245×10^{-10}
b	2.54
c	4.95

Table 5.2 Algorithm parameters for NAO

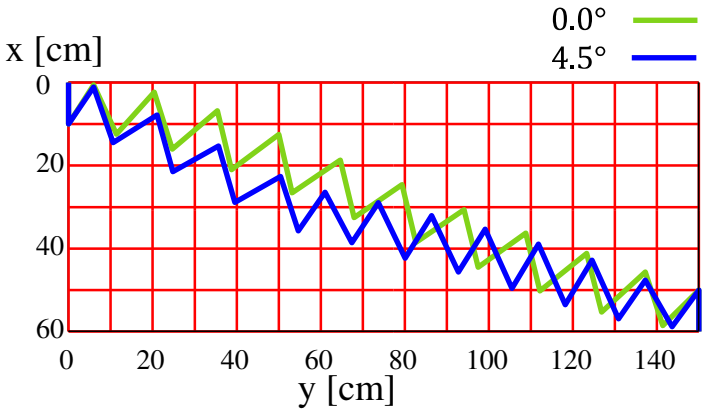
parameters	value
Δk_u	10^{-5}
Δk_{u_z}	10^{-5}
$\varepsilon_{\text{relax}}$	10^{-15}
$\alpha_{\min}$	0.15
β	0.5
γ	0.95
$\varepsilon_{\text{goal}}$	$(0.05, 0.05, 0.05, 0.05, 0.05, 0.05)^T$ (cm, deg)
$\varepsilon_{\text{input}}$	$(\ldots, 10^{-6}, 10^{-6}, 10^{-6}, 10^{-6}, 10^{-6}, 10^{-6}, \ldots)^T$ (cm, deg)
$\varepsilon_{\text{reachable}}$	$(2.5, 2.5, 0.5, 1.0, 0.5, 0.5)^T$ (cm, deg)
$n_{\text{reachable1}}$	10
$n_{\text{reachable2}}$	$3k$

Table 5.3 Map parameters for NAO

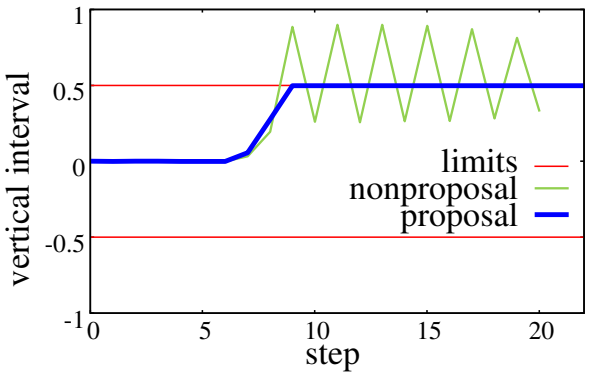
parameters	value
D_{grid}	5.0 cm
D_{sample}	1.0 cm



(a) x-y-z view



(b) x-y view



(c) vertical interval

Fig. 5.1 Footstep (slope)

Table 5.4　Result of planning (slope)

θ_{slope} (deg)	k_{init}	k	iteration	time(ms)
0.0	11	11	10	10.84
4.5	11	12	52	65.20

5.2　スロープ型の地形でのシミュレーション

初期状態 $\boldsymbol{q}^0 = (0.0, 0.0, 0.0, 10.0, 0.0, 0.0)^T$ (cm, deg) から目標状態 $\boldsymbol{q}_{\text{goal}} = (50.0, 150.0, 0.0, 10.0, 0.0, 0.0)^T$ へと向かう計画を行う．地形としては，y 軸方向において平面と傾斜角 θ_{slope} の斜面が $y = 50\,\text{cm}$ の位置で接続しているスロープ型の地形を用いる．

$$z_{\text{field}}(x,\ y) = \begin{cases} 0.0 & (y < 50.0) \\ (y - 50.0)\tan\theta_{\text{slope}} & (50.0 \leq y) \end{cases} \tag{5.1}$$

$\theta = 0.0°$，$4.5°$ の 2 パターンで計画を行う．

Figure 5.1，Table 5.4 に計画結果を示す．$4.5°$ の場合，平面上での歩幅に対して斜面上では歩幅を小さく取り，高低差制約を守りながら進む計画になっているのが Fig. 5.1c からも分かる．歩幅が狭いためステップ数は初期値から 1 増やす必要があり，また反復回数，計算時間は $0.0°$ の場合に比べていずれも増加している．

5.3　実地形データを用いた地形情報構築と計画

実際のロボットが自律移動を行う場合，環境の設計図などから事前に得られた地形形状を用いるのではなく，環境中で測域センサなどを用いて計測したデータから適宜地形情報を構築し，脚配置計画を行う必要がある．

ここでは，Kinect v2 を用いて実際に取得した点群データから地形情報を構築し，その地形データを用いて計画シミュレーションを行う．5.2 節と同様のスロープ型地形を用意し，ワールド座標系基準で (40.0, −113.0, 88.5) の位置にチルト角 $15.4°$ で見下ろす姿勢で設置した Kinect v2 で距離画像を取得する．そして，取得した画像をワールド座標系基準の点群データへ変換し，それを用いて 3.6 節に示す手順で $z_{\text{field}}(x, y)$ を構築する．問題としても 5.2 節と同様，初期状態 $\boldsymbol{q}^0 = (0.0, 0.0, 0.0, 10.0, 0.0, 0.0)^T$ (cm, deg) から目標状態 $\boldsymbol{q}_{\text{goal}} = (50.0, 150.0, 0.0, 10.0, 0.0, 0.0)^T$ へと向かう計画を行う．

Figure 5.2 および Table 5.5 に結果を示す．ワンスキャンでの計測であること，およびノイ

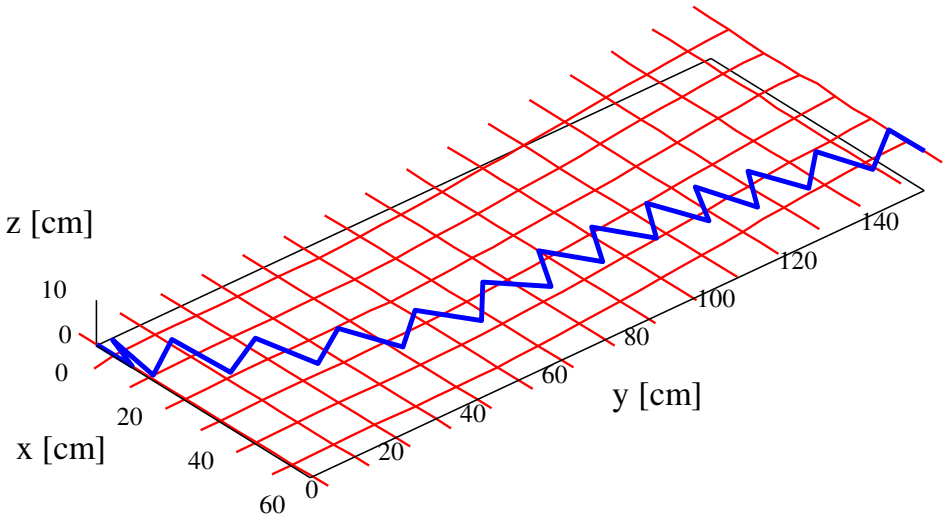


Fig. 5.2　Footstep (slope mapped by using Kinect v2)

Table 5.5　Result of planning (slope using map data from Kinect v2)

θ_{slope} (deg)	k_{init}	k	iteration	time(ms)
4.5	11	13	93	82.30

ズを考慮した地形情報構築ができていないことなどから，マップに歪みが含まれている．そのため，初期状態付近で地形適応制約を守るために小さい歩幅で歩行する必要があり，理想関数上での計画結果とは異なり 13 ステップでの計画となっている．計算時間は十分歩行周期以内に収まっており，実時間制は保たれている．

5.4　実機実験

ここからは，実機を用いた実験について述べる．

NAO への歩行指令では，まず脚配置計画を行い，得られた脚配置の系列から踏み出し量の系列を求める．次に，NAO の制御システムである NAOqi OS が標準で提供している，踏み出し量と時間を指定して踏み出しを行わせる API を歩数分だけ呼び出して歩行を指令する．この API 呼び出しに対して，NAOqi OS 内の歩行制御モジュールによって軌道生成が行われ，指示した脚配置計画を実現する歩行制御が行われる．なお，この API は平面上の歩行のためのものであり，平面上での位置と姿勢を指令するもので，内部で行われる歩行制御も平面歩行のためのものである．また，内蔵カメラの画像を用いた自己位置および姿勢の推定は行われておらず，外部のカメラによる計測や補正も行っていない．

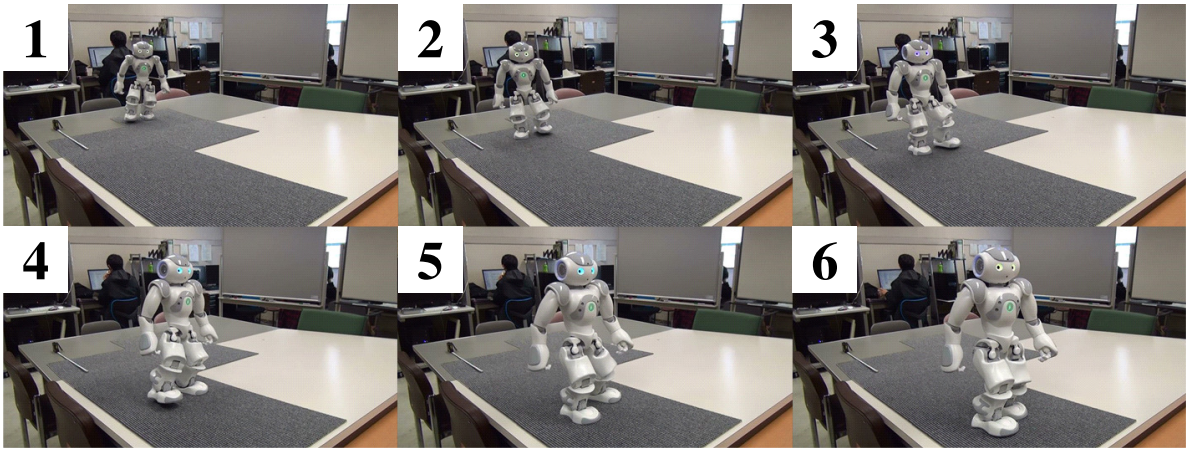


Fig. 5.3　Experiment (slope)

5.4.1　計画の実機適用と動作確認

まず，提案手法によって得られる脚配置計画が実際に実機に適用可能であることを確認する実験を行った．5.2 節におけるシミュレーションの傾斜角 4.5° の場合の計画を NAO に適用した．Figure 5.3 にその結果を示す．

平面上の歩行を仮定した歩行制御を用いて斜面を歩かせているため，歩行が不安定になり姿勢が計画からずれたものとなった．しかし，地形制約により上下方向の移動が NAO の許容値である 0.5 cm 以下に抑えられた計画がなされているため，制御のロバスト性で姿勢を維持でき，目標状態付近まで到達することができている．これにより，提案手法で実機に適用可能な脚配置計画が得られていることが確認できた．

5.4.2　目標到達精度および歩行の安定性の評価

次に，終了時の位置・姿勢と，歩行中の足裏圧力センサの値を計測し，目標状態への到達精度と，歩行の安定性について評価する実験を行った．いずれの評価においても，5.2 節における傾斜角 4.5° の場合の地形を対象とし，提案手法による地形形状を考慮した計画と，従来手法による平面上での計画の結果それぞれを用いて歩行させ，データを計測した．なお，提案手法を用いない場合は 11 ステップの計画となっており，提案手法における計画結果の 12 ステップより少ない．

まず，目標状態への到達精度の評価について述べる．歩行終了時の左脚の位置 x, y および姿勢 θ を計測し，計画で設定した目標状態 ($x = 50\,\mathrm{cm}$, $y = 150\,\mathrm{cm}$, $\theta = 0°$) に対する誤差を評価した．なお，初期状態は ($x = 0\,\mathrm{cm}$, $y = 0\,\mathrm{cm}$, $\theta = 0°$)．提案手法，従来手法それぞれ

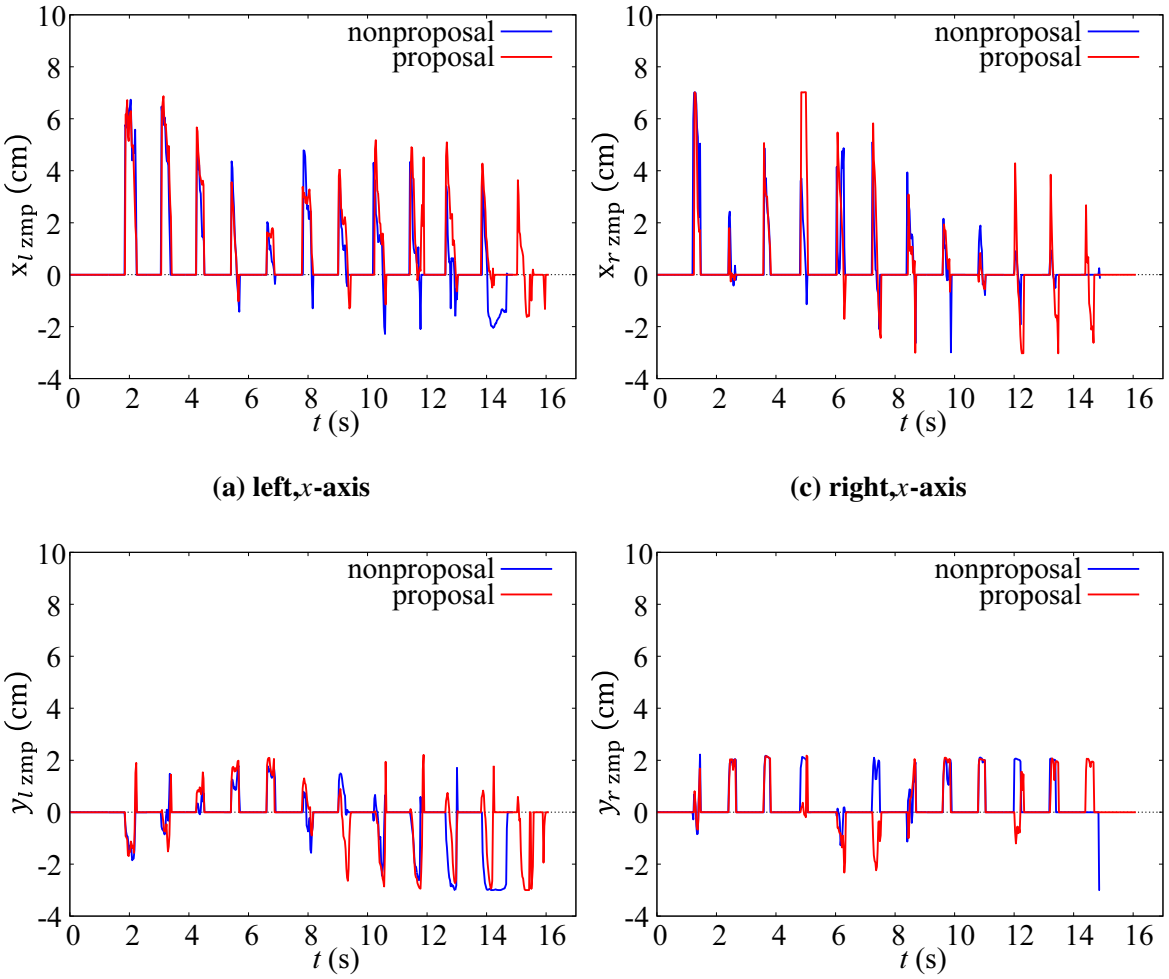
Table 5.6　Results of evaluation experiment of goal reaching error

mathod	x (cm)	y (cm)	θ (deg)
proposal	−24.4 ± 2.5	5.3 ± 1.0	21.0 ± 2.7
nonproposal	−46.5 ± 10.3	5.9 ± 3.3	39.1 ± 7.4
proposal (plane)	2.3 ± 3.7	6.7 ± 1.6	5.4 ± 2.8

で五回ずつ実験を行った．なお，NAO による歩行精度の参考として，提案手法での計画を用いて平面上を歩かせた場合の歩行についても同様に計測を行った．Table 5.6 に，提案手法および従来手法での斜面歩行における誤差および提案手法での平面歩行における誤差の，五回の試行での平均と標準偏差を示す．まず評価の前提として，提案手法で平面を歩行させた場合の結果から NAO による歩行では平面上においても誤差が生じてしまうこと，また提案手法での斜面歩行の結果と比較すると x，θ について誤差が増加しており斜面歩行では更に精度が落ちることがわかる．これを踏まえても，提案手法と従来手法とによる斜面歩行の結果を比較すると，いずれの要素についても明らかに誤差が改善されており，特に x，θ においては大きな改善が見られる．標準偏差を比較しても，いずれの要素においても提案手法のほうが小さく抑えられている．これらの結果から，従来手法に対して，提案手法によって目標到達の精度が改善されていることが分かる．

次に，片脚支持期の ZMP 値を用いた歩行の安定性の評価について述べる．NAO の左右の足裏それぞれ四ヶ所（(7.025, 2.310)，(7.025, −2.990)，(−3.025, 1.910)，(−2.965, −2.990)(cm)）に配置されている圧力センサの値を 25 ms ごとに計測し，脚先座標系 Σ_{fl} および Σ_{fr} 基準の片脚支持期の ZMP 応答値を算出した．Figure 5.4 に提案手法および従来手法それぞれにおける左右の x，y 軸方向の ZMP 応答値を示す．提案手法と従来手法とで 12 s 以降の左脚の ZMP 応答を比較する．x 軸方向 Fig. 5.4a 及び y 軸方向 Fig. 5.4b いずれについても，従来手法においては，本来 12 s 以前の応答と同様にすぐに 0 に戻るはずであるが，0 から変化したあとに戻るのが明らかに遅く，歩行が不安定になっていることが分かる．提案手法ではこのような ZMP の応答は見られず，それ以前と同様の応答を示しており，斜面上においても安定して歩行ができていることが分かる．

最後に，床反力の評価について述べる．ZMP 評価と同様に三回目の実験データを用い，左右それぞれについて圧力センサの示す値の合計値を比較した．一つの圧力センサで最大 25 N まで計測でき，片脚全体の床反力は最大 100 N である．Figure 5.5 に提案手法および従来手法それぞれにおける左右の足裏の床反力の遷移を示す．提案手法を用いない場合の結果において，左脚の床反力値が 12 s 以降に高い値をとっている．これはバランスを崩したことによって左脚が強く着地してしまったことを表しており，歩行が不安定になっていたと考えられる．これに対して，提案手法では床反力は最後まで周期的に変化しており，歩行の安定性が改善されていると言える．



(a) left,x-axis **(c) right,x-axis**

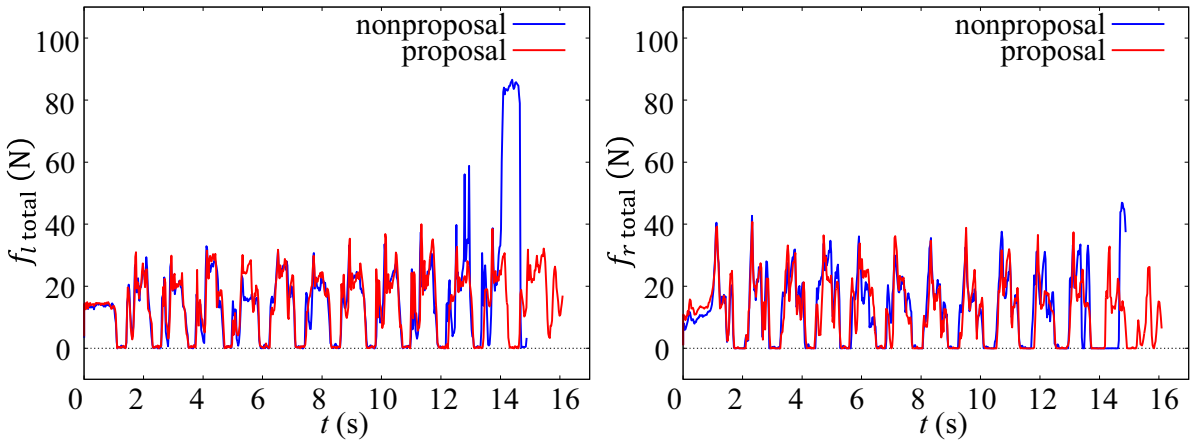
(b) left,y-axis **(d) right,y-axis**

Fig. 5.4　ZMP values in the single support phase

以上の実験データおよび評価から，目標到達精度，および ZMP 応答および床反力で評価した歩行の安定性ついて提案手法の有効性が確認された．

5.5　本章のまとめ

本章では，二足歩行ロボット NAO において，標準の平面歩行制御における垂直方向のロバスト性を利用した三次元地形での計画として，まず，計画シミュレーションによりスロープ形状の地形に適応した脚配置計画が得られることを確認した．また，Kinect v2 センサを用いて実際に計測したスロープに対しても同様に計画ができることを示した．さらに，実機実験により実際に前述のロバスト性を利用してスロープを歩行できる計画が得られているこ



(a) left **(b) right**

Fig. 5.5　Ground reaction force

とや，目標到達の精度や歩行の安定性について有効性があることを示した．

第 6 章

結論

6.1 まとめ

本論文では，二足歩行ロボットにおける地形形状を考慮した三次元脚配置計画の手法を提案した．従来の水平方向の脚配置計画のためのモデルに，独立した垂直方向の動作を追加し，脚配置遷移モデルの状態および入力についての次元の拡張を行うことなく，事前に与えられた地形情報に基いて垂直方向の移動量を制限する制約を定式化し追加することで，計算コスト増加を抑えつつ三次元地形への適応を可能にした．ヒューマノイドロボット HRP-2 を想定した計画シミュレーションを行い，連続的変化の三次元地形に対しては実時間で脚配置計画の最適化が行えることを示した．不連続な変化のある地形では計算が不安定化してしまうといった課題が残るが，地形情報の構築方法や計算手法などの見直しで改善できると考えられる．また，ヒューマノイドロボット NAO を想定した計画シミュレーションも行い，HRP-2 想定の時と同様に実時間で地形適応を実現したかいが得られることを示し，また実機実験により，実際に二足歩行ロボットに適用可能な脚配置が得られていることを確認した．

6.2 課題と展望

残された課題として，まず計画シミュレーションにおいても示された不連続な地形変化に対する求解性能の低下を改善する必要がある．現在の二次計画法を利用した反復法による最適化は，[19] でモデル予測制御に使われていることなどから実用上は十分なものであるが，その収束性については厳密には保証されていない [22]．また，脚先の三次元的な姿勢および形状を考慮できていないという問題がある．これらを考慮して制約条件を追加し，地形との接触に関して傾きの大きすぎる場所への接地を避けるような計画を行う必要がある．これに関連して，地形情報の構築方法についても，現在の簡易的なものではなく，誤差を考慮した

制約設定を可能にする，実際の形状からの誤差範囲を保証できるような方法の検討が必要である．また，三次元歩行における消費エネルギを評価に含めて最適化を行う計画手法について検討を進めていく．これは，現在の近似前の評価関数は目標状態との誤差のみを評価しており，ロボットの移動において重要な指標となる消費エネルギを最適化できていないためである．

将来的な展望として，モデルベースと探索手法の住み分けについての検討が必要であると考えている．提案手法の目標距離の限界は原理上存在しないが，実際の制御では脚配置に誤差が必ず発生するため長距離に渡る最適化を行うのは実用上の意味があまりないと考えられる．そこで，離散的な粗い探索による長距離の歩行計画と，提案手法であるモデルベースの脚配置最適化を組み合わせた手法が必要になると考えており，組み合わせ方や住み分けの仕方について調査・検討を進めていく．また，現状の歩容制御と切り離した上での脚配置計画ではなく，歩行制御計画と脚配置計画とを統合した，歩容全体の計画を行う手法の検討を進めていく．

謝辞

本論文を作成するにあたり，終始懇切なるご指導，御鞭撻を賜りました本学電子情報系専攻の田窪朋仁教授に厚く御礼申し上げます.
また，研究を進める上で様々なご助言をいただきました，上野敦志講師に感謝の意を表します.
最後になりましたが，本研究室の皆様にも日頃からご協力頂いたことに感謝いたします.

参考文献

[1] Jwu-Sheng Hu, Jung-Hung Cheng and Yung-Jung Chang, “Spatial Trajectory Tracking Control of Omni-directional Wheeled Robot Using Optical Flow Sensor”, in Proc. of 2007 IEEE Int. Conf. on Control Applications (CCA2007), pp. 1462–1467, 2007.

[2] Elie Maaloufa, Maarouf Saada and Hamadou Saliahb, “A Higher Level Path Tracking Controller for a Four-wheel Differentially Steered Mobile Robot”, Robotics and Autonomous Systems. vol. 54, issue 1, pp. 23–33, 2006.

[3] James J. Kuffner Jr., Koichi Nishiwaki, Satoshi Kagami, Masayuki Inaba and Hirochika Inoue, “Footstep Planning Among Obstacles for Biped Robots”, in Proc. of 2001 IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS2001), vol. 1, pp. 500–505, 2001.

[4] Ryo Kurazume, Tsutomu Hasegawa and Kan Yoneda, “The Sway Compensation Trajectory for a Biped Robot”, in Proc. of 2003 IEEE Int. Conf. on Robotics and Automation, vol. 1, pp. 925–931, 2003.

[5] Koichi Nishiwaki, Satoshi Kagami, James J. Kuffner, Masayuki Inaba and Hirochiaka Inoue, “Online Humanoid Walking Control System and a Moving Goal Tracking Experiment”, in Proc. of 2003 IEEE Int. Conf. on Robotics and Automation (ICRA2003), vol. 1, pp. 911–916, 2003.

[6] Hidehito Kobayashi and Tomomichi Sugihara, “Self-consistent Automatic Navigation of COM and Feet for Realtime Humanoid Robot Steering”, in Proc. of 2009 IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS2009), pp. 3525–3530, 2009.

[7] Joel Chestnutt, Manfred Lau, German Cheung, James Kuffner, Jessica Hodgins and Takeo Kanade, “Footstep Planning for the Honda ASIMO Humanoid”, in Proc. of 2005 IEEE Int. Conf. on Robotics and Automation (ICRA2005), pp. 629–634, 2005.

[8] Johannes Garimort, Armin Hornung, Maren Bennewitz, “Humanoid Navigation with Dynamic Footstep Plans”, in Proc. of 2011 IEEE Int. Conf. on Robotics and Automation (ICRA2011), pp. 3982–3987, 2011.

[9] Armin Hornung, Maren Bennewitz, “Adaptive Level-of-Detail Planning for Efficient Humanoid Navigation”, in Proc. of 2012 IEEE Int. Conf. on Robotics and Automation

(ICRA2012), pp. 997–1002, 2012.

[10] Weiwei Huang, Junggon Kim and Christopher G. Atkeson, “Energy-based Optimal Step Planning for Humanoids”, in Proc. of 2013 IEEE Int. Conf. on Robotics and Automation (ICRA2013), pp. 3109–3114, 2013.

[11] Hong Liu, Qing Sun and Tianwei Zhang, “Hierarchical RRT for Humanoid Robot Footstep Planning with Multiple Constraints in Comples Environments”, in Proc. of 2012 IEEE/RSJ Int. Conf on Intelligent Robots and Systems (IROS2012), pp. 3187–3194, 2012.

[12] Oussama Kanoun, Eiichi Yoshida and Jean-Paul Laumond, “An Optimization Formulation for Footsteps Planning”, in Proc. of 2009 9th IEEE-RAS Int. Conf. on Humanoid Robots (Humanoids2009), pp. 202–207, 2009.

[13] Oussama Kanoun, Jean-Paul Laumond and Eiichi Yoshida, “Planning Foot Placements for a Humanoid Robot: A Problem of Inverse Kinematics”, International Journal of Robotics Research, vol. 30, issue 4, pp. 476–485, 2011.

[14] Robin Deits and Russ Tedrake, “Footstep Planning on Uneven Terrain with Mixed-Integer Convex Optimization”, in Proc. of 2014 14th IEEE-RAS Int. Conf. on Humanoid Robots (Humanoids2014), pp. 279–286.

[15] Robin Deits and Russ Tedrake, “Computing Large Convex Regions of Obstacle-Free Space Through Semidefinite Programming” Algorithmic Foundations of Robotics XI, Springer International Publishing, pp. 109–124, 2015.

[16] Nobuya Yao, Tomohito Takubo, Kenichi Ohara, Yasushi Mae and Tatsuo Arai, “Gait Planning for a Biped Robot by a Nonholonomic System with Difference Equation Constraints”, in Proc. of 2010 IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS2010), pp. 4471–4476, 2010.

[17] 八百伸弥, 田窪朋仁, 大原賢一, 前泰志, 新井健生, “差分方程式拘束を伴う離散時間非ホロノミック系を用いた 2 足歩行計画の最適化”, 第 16 回ロボティクス・シンポジア, pp. 249–254, 2011.

[18] Donald Goldfarb and A. Idnani, “A Numerically Stable Dual Method for Solving Strictly Convex Quadratic Programs”, Mathematical Programming, vol. 27, issue 1, pp. 1–33, 1983.

[19] Jan M. Maciejowski, “Predictive Control with Constraints”, 足立修一（訳）, 管野政明（訳）, 東京電機大学出版局, 2005.

[20] Tomomichi Sugihara, “Solvability-Unconcerned Inverse Kinematics by the Levenberg-Marquardt Method”, IEEE Transactions on Robotics, vol. 27, issue 5, pp. 984–991, 2011.

[21] 杉本博之, “制約最適化問題のスケーリングについて”, 土木学会論文集, vol. 356, pp. 579–582, 1985.

[22] 矢部博, 八巻直一, “非線形計画法”, 朝倉出版, 1999.

[23] 五十棲隆勝, 赤地一彦, 平田勝, 金子健二, 梶田秀司, 比留川博久, “ヒューマノイドロボット HRP-2 の開発”, 日本ロボット学会誌, vol. 22, no. 8, pp. 1004–1012, 2004.

[24] Aldebaran Robotics, “Nao robot: characteristics - Aldebaran”, https://www.aldebaran.com/en/cool-robots/nao/find-out-more-about-nao (2016/2/21 アクセス確認).

[25] Aldebaran Robotics, “KEY FEATURE (Stable and Omni-directional Walk)”, https://www.aldebaran.com/sites/aldebaran/files/featurepaperstableandomnidirectionalwalk.pdf (2016/2/21 アクセス確認).

[26] Aldebaran Robotics, “Locomotion control — Aldebaran 2.1.4.13 documentation”, http://doc.aldebaran.com/2-1/naoqi/motion/control-walk.html (2016/2/21 アクセス確認).

[27] Microsoft, “Kinect hardware”, Kinecthardwarehttps://dev.windows.com/en-us/kinect/hardware (2016/2/21 アクセス確認).

研究業績

国内発表（査読なし）

- 小林大気, 田窪朋仁, 上野敦志, “離散時間運動学モデルに基づく二足歩行ロボットの 3 次元脚配置計画”, ロボティクス・メカトロニクス講演会 2013 (ROBOMEC2013), 1A1-P11, May 2013.

国際発表（査読あり）

- **Daiki Kobayashi**, Tomohito Takubo, Atsushi Ueno, “3D Gait Planning Based on Discrete-time Kinematic Model of Biped Walking”, The 25th 2014 International Symposium on Micro-Nano Mechatronics and Human Science (MHS2014), Nagoya, Japan, Nov. 2014.

論文誌

- **Daiki Kobayashi**, Tomohito Takubo, Atsushi Ueno, “Model-Based Footstep Planning Method for Biped Walking on 3D Field”, Journal of Robotics and Mechatronics, vol. 27, no. 2, pp. 156–166, Apr. 2015.