学術・技術論文

実機の物理的制約を考慮した即応的脚動作生成手法

金広文男* 森澤光晴* Suleiman Wael*
金子健二* 吉田英一*

Reactive Leg Motion Generation Method under Consideration of Physical Constraints

Fumio Kanehiro*, Mitsuharu Morisawa*, Wael Suleiman*,
Kenji Kaneko* *and* Eiichi Yoshida*

This paper proposes a reactive leg motion generation method which integrates physical constraints into its generation process. In order to react given instructions instantaneously or to keep balance against external disturbances, feasible steps must be generated automatically in real-time for safety. In many cases this feasibility has been realized by using predefined steps or admissible stepping regions. However, these predefinitions are valid only in limited situations. The proposed method considers physical constraints during its generation process. It consists of a swing leg trajectory generator and a constraint solver. The former generates a swing leg trajectory considering rough self-collision avoidance between legs. The latter does joint velocities considering joint angle/velocity limits and precise self-collision avoidance. Moreover, in order to improve the possibility of feasible patterns being generated, a stiffness varying constraint and a landing position modification function are introduced. The proposed method is validated by experiments using a humanoid robot HRP-2.

Key Words: Optimization, Collision Avoidance, Motion Planning, Biped Walking

1. 緒　　言

ある動作パターンが実際のロボットで実行可能なものであるためには，その動作パターンはロボットに与えられたタスクを完遂するための必要条件だけではなく，関節可動範囲や速度範囲，自己干渉回避等の物理的制約も満たしたものでなければならない．しかし多くの場合，これらの物理的制約のいくつかは動作パターン生成の過程では考慮されず，動作パターン生成器に入力するパラメータの範囲を制限したり，生成後に実現可能であるかを検査して自動あるいは手動で実現可能となるように修正したりしているのが現実である．

文献 [1] で紹介されている動作パターンデザインツールは関節可動範囲や速度範囲の条件が満たされていない場合，その箇所は提示するが修正はデザイナーの手に委ねている．これはオフラインでの動作生成が前提であり，デザイナーのイメージにできるだけ近い動作が得られることを目的としているためであると考えられる．文献 [2] はモーションキャプチャデータをヒューマノイドの機構に適用した関節角軌道を階層型 B スプラインによって表現し，その係数を物理的制約下で最適化することで実現可能な軌道に変換する手法を提案している．これもオフラインでの動作生成が前提の手法である．

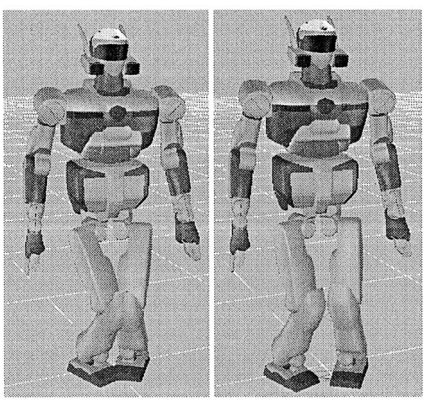

Fig. 1 Example postures generated without (left) and with (right) considering physical constraints

一方，ジョイスティック等の操作デバイスからの入力に応じて動作を生成したり，バランスを崩した際にロボットが自律的に踏み出したりするような即応的な動作生成の場合には物理的制約を満たした動作が自動的，かつ短時間で生成されることが求められる．そこで本稿では **Fig. 1** 左のような自己干渉が発生

原稿受付 2010 年 3 月 19 日
*産業技術総合研究所
*National Institute of Advanced Industrial Science and Technology (AIST)
■ 本論文は有用性で評価されました．

するような歩行動作指令が与えられた場合でも，Fig. 1 右のような物理的制約を満たした動作をオンラインで生成する脚動作生成手法を提案する．

オンラインで物理的制約を満たした歩行動作を生成するためのアプローチの一つは事前に物理的制約を満たす動作を生成できる着地位置の条件を検査しておき，オンラインではその条件下で着地位置を選択する方法である．文献 [3] は次の着地位置をあらかじめ用意したいくつかの着地位置に制限し，これを歩行動作計画における探索ツリーの拡張に用いている．着地点の種類は探索時間との兼ね合いもあり少数に抑えられているが，これは同時にロボットが持っている能力を制限することにもなる．これに対して文献 [4] は次の着地位置を少数の点ではなく領域として与える方法を提案している．着地可能な領域をオフラインで検査しておき，この領域外に次の一歩が指定された場合には領域内に収まるように着地位置を変更する方法である．この着地可能な領域は両足先の位置関係だけでなく，腰の高さ等様々な条件によって左右されるため利用可能な状況が限定される問題がある．

もう一つのアプローチは物理的制約を逆運動学解法に組み込んで動作生成を行うものであり，本稿で提案する手法もこのアプローチに属する．ロボットに与えられたタスクがエンドエフェクタ等ロボットのいくつかの部位に対する拘束条件として表現できる場合，それらの拘束条件群を満たすロボットの関節角度列は逆運動学解法を用いて計算される．コンピュータグラフィクスの分野では自然な動作を，ロボティクスの分野では安全な動作を生成するために関節可動範囲を考慮した逆運動学解法がいくつか提案されている．例えば文献 [5] は複数の優先度付拘束条件を可動範囲を満たしながら解く方法を提案している．同様の枠組みに物体間の距離に基づく線形等式拘束条件を追加することで干渉回避を行う手法が提案されている [6] [7] が，本来物体間の距離に対する拘束は距離が一定値以上であればよいという不等式拘束で表されるため，等式拘束は必要以上に拘束が強く，他の拘束条件が満たせなくなる可能性が大きくなる問題がある．またこれらの研究では腕部の干渉回避を題材として取り扱っており，歩行動作の生成において問題となる，脚の少なくとも一方が常に接地して地面に拘束されていること，脚同士の干渉を回避するための動作が適切でない場合には脚同士が絡まった状態となり歩行が継続できなくなること等には触れていない．

文献 [8] は高速な歩行パターン生成手法 [9] と凸包によって近似したリンク間の最短距離を V-Clip [10] を使って追跡する高速なリンク間の干渉検査手法を組み合わせ，干渉の発生が予測される場合にはあらかじめ生成してあった安全に停止できるパターンと接続することで安全に歩行動作を停止する方法を提案している．この方法では必ず安定に停止できることが保証できるが行動を継続することはできず，本研究とは目的が異なる．

本稿では物理的制約を満たした脚動作をオンラインで生成可能な動作生成手法を提案する．ここで物理的制約条件としては，関節可動範囲，関節速度範囲，関節トルク範囲，動力学的バランスの維持，自己干渉の回避が考えられる．これらのうち，本稿で提案する手法では，自己干渉回避，関節可動範囲，関節速度範囲に関する拘束条件を考慮する．動力学的バランスに関してはロボットを質点とみなした場合のモデル化誤差によって生じる ZMP [11] 軌道の参照軌道からの変動が十分に小さいと仮定して脚の運動に対する精密なバランス補償は行わない．この仮定が非現実的なものでないことを 6 章で確認する．また関節トルク範囲に関しては本稿では考慮しない．これは後述するように提案手法では各種制約条件が関節速度に対する線形等式，不等式として与えられることが必要であるためである．次に「オンラインで生成可能」とは計算時間が対象システムで許容できる範囲内であるというだけでなく，次の着地位置がロボットの制御周期ごとに変更された場合でも，それに対応した動作生成が可能であることを意味する．実機で実現可能な動作を生成するため，提案する手法では物理的制約を動作生成の過程に組み込む．具体的には物理的制約を関節速度に対する線形等式，不等式拘束として表現してその制約下で必要な脚動作を遂行するのに最適な関節速度を計算するローカルな動作生成手法を基礎とし，解と成りうる関節速度の範囲を広げるための厳密度可変拘束，脚同士の交差を防ぐための足部の位置関係を考慮した遊脚軌道生成器，着地位置を到達可能なものに修正するための着地位置補正機能を組み合わせたものである．

本稿は以下のように構成されている．2 章では本稿で実現する 2 足歩行パターン生成器の概要について述べる．3 章では物理的制約を含む複数の制約条件下で最適な関節速度を計算し，実現可能な動作生成を行う動作生成手法について述べる．4 章では前章で述べた手法の欠点を補うための遊脚軌道生成手法について述べる．5 章では実現可能な動作生成ができる範囲を広げるため，着地位置の補正が行えるように提案手法を拡張する．6 章で実験により提案手法の有効性を確認し，7 章で本論文をまとめる．

2. 2 足歩行パターン生成器の概要

本稿で実現する 2 足歩行動作パターン生成器の全体構成を **Fig. 2** に示す．

2 足歩行動作パターン生成器に入力されるのは，着地位置の列である．この着地位置はオペレータがジョイスティック等の操作デバイスを操作したり，文献 [3] のような歩行経路計画アル

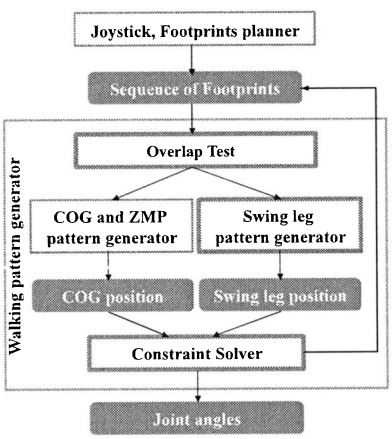

Fig. 2 Overall structure of walking pattern generator

ゴリズムが生成したり，バランス制御モジュールがバランスを維持するために有効な着地位置を生成したりすることで与えられる．

入力された着地位置列は 2 足歩行パターン生成器の中核である重心軌道生成器と遊脚軌道生成器とに入力される．本稿では次の着地位置の変更に即時に対応可能で，重心軌道と ZMP 軌道を同時に生成可能なアルゴリズム [12] に基づく重心軌道生成器を用いている．これらの生成器が計算した重心位置および遊脚の位置を実現する関節角度列を制約解決器が計算する．（ただし本稿では，重心位置を直接拘束条件として用いるのではなく，重心位置に一定のオフセットを加えたものを腰の位置に対する拘束条件として用いている．）なお，この 2 足歩行パターン生成器内の処理は着地位置列が入力された際に一度だけ実行されて一連の歩行パターンが一括して出力されるのではなく，制御周期毎に実行されてその時刻にとるべき関節角度列だけを計算する．これは着地位置の変更に即応的に対応するためである．

この全体構成そのものは従来の 2 足歩行パターン生成器と同じである．物理的制約を満たしていない動作パターンが生成される原因は，解析的逆運動学解法など物理的制約を考慮しない手法が制約解決器に使われること，遊脚の現在位置と次の着地位置を水平面内においては直線で結ぶような脚同士の干渉を考慮しないアルゴリズムが遊脚軌道生成器に使われること，実現不可能な着地位置が指令されること等にある．そこで本稿では物理的制約を考慮した制約解決器を 3 章で，遊脚軌道生成器を 4 章で，着地位置の補正機能を 5 章で導入する．

3. 制 約 解 決 器

重心軌道生成器から生成された腰の位置・姿勢と遊脚軌道生成器から生成された遊脚の位置・姿勢を満たすために必要な関節速度 $\dot{\boldsymbol{q}}$ は以下の連立一次方程式を解くことで得られる．

$$\begin{aligned} \boldsymbol{J}_{waist}\dot{\boldsymbol{q}} &= \dot{\boldsymbol{x}}_{waist}, \\ \boldsymbol{J}_{foot}\dot{\boldsymbol{q}} &= \dot{\boldsymbol{x}}_{foot} \end{aligned} \tag{1}$$

ここで $\boldsymbol{J}_{waist}$ は支持脚先端からみた腰位置・姿勢のヤコビアン，$\boldsymbol{J}_{foot}$ は支持脚先端からみた遊脚先端の位置・姿勢のヤコビアン，$\boldsymbol{x}_{waist}, \boldsymbol{x}_{foot}$ はそれぞれ重心軌道生成器，遊脚軌道生成器が計算した腰および遊脚先端の目標位置・姿勢である．冗長な自由度が存在する場合には解が無数に存在するため，擬似逆行列を用いてノルム最小となる関節角度の変位を求めるのが一般に広く用いられている手法であり，これは次の最適化問題を解いているのに等しい（ただし式（1）を満たす $\dot{\boldsymbol{q}}$ が存在しない場合，最適化問題（2）は解を持たないため結果は異なる）．

$$\begin{aligned} &\min_{\dot{\boldsymbol{q}}} \|\dot{\boldsymbol{q}}\|^2 \\ \text{subject to} \quad &\boldsymbol{J}_{waist}\dot{\boldsymbol{q}} = \dot{\boldsymbol{x}}_{waist}, \\ &\boldsymbol{J}_{foot}\dot{\boldsymbol{q}} = \dot{\boldsymbol{x}}_{foot}. \end{aligned} \tag{2}$$

本稿では物理的制約を満たすために必要な条件を線形等式，不等式条件として表現してこの最適化問題に挿入することで物理的制約を満たした関節速度を求め，これを用いて関節角度を更新する処理を制御周期毎に繰り返すことで動作生成を行う．この方法は脚動作に限定されるものではなく，様々な動作生成一般に適用可能なものである [13]．物理的制約として以下のものを考慮する．

（1）自己干渉の回避
（2）関節可動範囲の制約
（3）関節速度範囲の制約

以下これらの物理的制約を線形等式，不等式条件として定式化する．

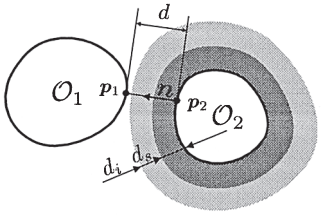

Fig. 3 Collision avoidance: Faverjon and Tournassoud's method

3.1 干渉回避拘束

干渉回避には Faverjon らによって提案された *velocity damper* を用いる．*velocity damper* とは二つの物体間の最短距離に対する以下のような拘束条件である [14].

$$\dot{d} \geq -\xi \frac{d - d_s}{d_i - d_s} \quad \text{if} \quad d < d_i \tag{3}$$

ここで d は二つの物体間の最短距離，ξ が減速の度合いを調整する正のパラメータ，d_i が *velocity damper* が有効となる距離の閾値，d_s が物体間に確保する最小距離である（**Fig. 3** 参照）．この拘束を適用することによって d が d_i よりも小さくなるにつれ，d を小さくする方向への速度に対する制限が徐々に厳しくなり，d は決して d_s 以下にはならない．

物体間の最近点を $\boldsymbol{p_1}, \boldsymbol{p_2}$ とし，$\boldsymbol{p_1}, \boldsymbol{p_2}$ を結ぶ方向の単位ベクトルを $\boldsymbol{n}$ とすると式（3）は以下のように書き換えられる．

$$\boldsymbol{n}^T(\boldsymbol{J}_{\boldsymbol{p}_1} - \boldsymbol{J}_{\boldsymbol{p}_2})\dot{\boldsymbol{q}} \geq -\xi \frac{d - d_s}{d_i - d_s} \tag{4}$$

ここで $\boldsymbol{J}_{\boldsymbol{p}_1}, \boldsymbol{J}_{\boldsymbol{p}_2}$ はそれぞれ $\boldsymbol{p_1}, \boldsymbol{p_2}$ のヤコビアンである．このように *velocity damper* は関節速度に対する線形不等式拘束として記述することができるため，動作生成のフレームワークに組み込むことができる．ただしこの拘束を用いて連続な関節速度を解として得るためには，$\boldsymbol{p_1}$，$\boldsymbol{p_2}$ は両物体の表面上を連続的に移動しなければならず，そのためには両物体は狭義に凸な形状を持った物体でなくてはならない．狭義に凸な形状とは，物体の表面または内部から任意の 2 点を選択した場合に，それらの 2 点を結ぶ線分上の端点を除くいかなる点も物体の表面または外部には存在しないような形状である．球や楕円体が狭義に凸な形状であるのに対して，直方体や円筒，凸包は凸な形状ではあるが狭義に凸な形状ではない．

3.2 関節角度・角速度範囲拘束

関節角度，角速度に関しても制約条件を満たしながら関節速度が不連続になることのないように，以下のような *velocity damper* を用いた拘束条件を各関節速度 $\dot{q}_j$ に対して定義する．

$$\dot{q}_j^{max}(q_j) \geq \dot{q}_j \geq \dot{q}_j^{min}(q_j), \text{ for } j \in \{1, ..., n_{dof}\} \tag{5}$$

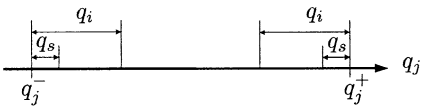

Fig. 4 Velocity damper for joint limits of the j-th joint

ここで n_{dof} は関節数である．$\dot{q}_j^{max}(q_j)$，$\dot{q}_j^{min}(q_j)$ の値は次式により決定する．

$$\dot{q}_j^{max}(q_j) = \begin{cases} \dot{q}_j^+ \dfrac{(q_j^+ - q_j) - q_s}{q_i - q_s} & \text{if } q_j^+ - q_j \le q_i, \\ \dot{q}_j^+ & \text{otherwise} \end{cases} \tag{6}$$

$$\dot{q}_j^{min}(q_j) = \begin{cases} \dot{q}_j^- \dfrac{(q_j - q_j^-) - q_s}{q_i - q_s} & \text{if } q_j - q_j^- \le q_i, \\ \dot{q}_j^- & \text{otherwise} \end{cases} \tag{7}$$

ここで q_j^+, q_j^- は関節角度の上下限値，$\dot{q}_j^+, \dot{q}_j^-$ は関節速度の上下限値であり，これらのパラメータはハードウェアの機械的・電気的仕様から決定する．q_i, q_s はそれぞれ，q_j^+, q_j^- からの関節速度が制約を受ける範囲，関節角度が決して侵入しない範囲の幅を表す（**Fig. 4** 参照）．q_j^+, q_j^- によって定義される機械的可動範囲全域を q_j が移動できるようにするため，通常は q_s には 0 を設定する．

3.3 厳密度可変拘束

ヒューマノイドは非常に多自由度なシステムではあるが，それでも体の何点かの軌道を与えて動作生成を行った場合，物理的拘束を満たすために使える自由度の数は多くはない．歩行動作の生成においては腰と両足先の軌道が与えられるのに対して両脚の自由度が合わせて 12 しかない場合，脚同士の干渉が発生してもそれを回避する為に使える自由度は 0 であり，干渉を回避することは不可能となる．しかしこの場合，遊脚が空中を移動する間は必ずしも指定された軌道を厳密にトレースする必要はないため，遊脚が参照軌道から位置をずらして干渉を回避することで実現可能な歩行動作を生成することが可能となる．このような仕組を実現するためには，式（1）に示した拘束条件のうちいくつかについて誤差の発生を許す必要がある．そこで，本来の等式条件に対して発生する誤差ベクトル $\boldsymbol{e}$ を導入し，誤差を許容する拘束条件に対する等式条件を次式のように与える．

$$\boldsymbol{J}_{relax}\dot{\boldsymbol{q}} + \boldsymbol{e} = \dot{\boldsymbol{x}}_{relax} \tag{8}$$

ここで $\boldsymbol{J}_{relax}$，$\dot{\boldsymbol{x}}_{relax}$ は誤差を許容する拘束条件に対応する行を $\boldsymbol{J}_{waist}, \boldsymbol{J}_{foot}, \dot{\boldsymbol{x}}_{waist}, \dot{\boldsymbol{x}}_{foot}$ から抜き出してまとめたものである．他の誤差が許容されない各種拘束条件を満たした上で誤差を最小限に抑えるため，$\tilde{\boldsymbol{q}} = [\dot{\boldsymbol{q}}^T \ \boldsymbol{e}^T]^T$ とおき，これを用いて最適化問題（2）を以下のように書き換える．

$$\begin{aligned} &\min_{\tilde{\boldsymbol{q}}} \tilde{\boldsymbol{q}}^T \boldsymbol{W} \tilde{\boldsymbol{q}} \\ \text{subject to} \quad &\tilde{\boldsymbol{J}}_{waist}\tilde{\boldsymbol{q}} = \dot{\boldsymbol{x}}_{waist} \\ &\tilde{\boldsymbol{J}}_{foot}\tilde{\boldsymbol{q}} = \dot{\boldsymbol{x}}_{foot} \end{aligned} \tag{9}$$

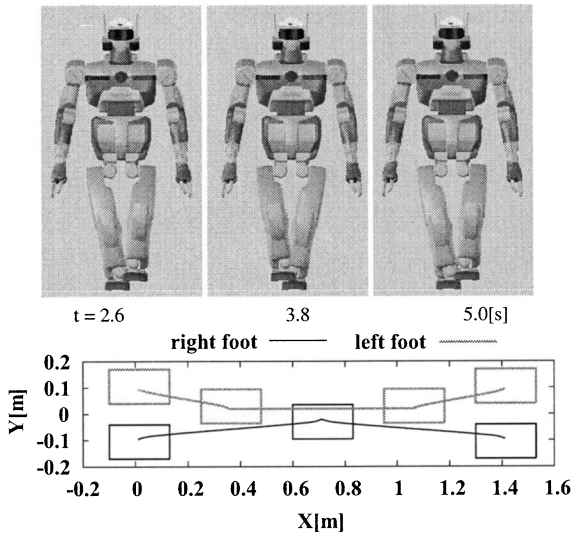

Fig. 5 Foot trajectories and postures generated by our conventional pattern generator. There are collisions between feet

ここで $\boldsymbol{W}$，$\tilde{\boldsymbol{J}}_{waist}$，$\tilde{\boldsymbol{J}}_{foot}$ は以下のように定義する．

$$\begin{aligned} \boldsymbol{W} &= \begin{bmatrix} \boldsymbol{I} & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{W}_e \end{bmatrix}, \\ \tilde{\boldsymbol{J}}_{waist} &= \begin{bmatrix} \boldsymbol{J}_{waist} & \boldsymbol{S}_{waist} \end{bmatrix}, \\ \tilde{\boldsymbol{J}}_{foot} &= \begin{bmatrix} \boldsymbol{J}_{foot} & \boldsymbol{S}_{foot} \end{bmatrix}. \end{aligned} \tag{10}$$

$\boldsymbol{W}_e$ は目的関数において誤差ベクトル $\boldsymbol{e}$ に対する重み付けを行う対角行列である．大きな重みを設定すると誤差によって目的関数が受けるペナルティが大きくなるために対応する誤差成分を小さくするような解が求められ，小さな重みを設定すると逆に大きな誤差も許容するような解が求められる．以下では，誤差が小さい場合を「厳密度が高い」，大きい場合を「厳密度が低い」と表現し，式（8）の形式で与えられる拘束条件を厳密度可変拘束と呼ぶ．ただし，誤差の値は目的関数内の重み付けによって相対的に決定されるため，誤差の絶対値によって厳密度は定義できない．$\boldsymbol{S}_{waist}$，$\boldsymbol{S}_{foot}$ は 12 の拘束条件から厳密度可変にする拘束条件を選択するための選択行列である．前述の歩行動作の生成例で言えば，遊脚先端位置の拘束条件式に対して厳密度可変拘束を適用し，遊脚期の開始時と終了時に重みを大きく，中間で小さくすることで，着地位置は維持しながら空中では干渉を避ける，といったことが可能となる．

3.4 動作生成例

これまでに述べた手法を用いることで，物理的制約を満たした動作パターンが生成されることを確認するため，自己干渉が発生する着地位置列を用いて動作生成を行った．動作生成の対象として用いたのはヒューマノイドロボット HRP-2 [15] である．

提案する手法を用いずに歩行動作を生成した場合の結果を **Fig. 5** に示す．上段が遊脚が支持脚の真横を通過する瞬間のロボットの姿勢，下段が着地位置および足先の軌道である．矩形は着地位置での足部の外形を表している．中間の 3 歩の左右方

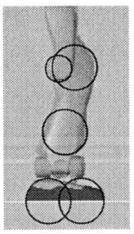

Fig. 6 Arrangement of spheres

向の距離を狭めてあり，通常の遊脚軌道生成器は二つの着地位置を水平面内においては直線で結ぶ軌道を生成するために脚同士が干渉してしまう．（遊脚の軌道が完全な直線でないのは，直線軌道を元に踵着地や爪先離陸等の足部の回転運動を加えているためである．）

自己干渉回避を行うためにはロボットの幾何情報を与える必要がある．ロボットを構成する物体は平面や曲面が複雑に組み合わされた形状をもっているが，この形状をそのまま取り扱うことは困難であるため，この形状を三角形パッチが貼り合わされた多面体で近似する手法が広く用いられている．しかし，前述のように連続な関節角速度を得るためには幾何情報は狭義に凸な形状でなければならないという条件があり，狭義に凸でない多面体で近似する方法は使えない．そのため球のような狭義に凸な形状を持ったプリミティブ形状を組み合わせる，球とトーラスのパッチを張り合わせる [16] といった方法で形状を近似するか，あるいは筆者らが提案した狭義に凸でない形状にも適用可能な干渉回避手法 [17] を用いる必要がある．後者は三角形パッチの集合として与えられた形状データをそのまま利用できるという利点があるが，数多くの拘束条件を解く必要があり，計算時間の限られたオンラインの動作制御には適用できない．そこで本稿では複数の球を組み合わせることでロボットの形状を近似する．本稿では脛のリンクを三つの球で，足部を二つの球で近似した．これらの球の配置およびサイズを **Fig. 6** に示す．本来のロボットの形状すべてが球で覆われているわけでなないため，それ以外の部分で干渉が発生する可能性はある．しかし，本来の形状すべてを覆うために球の数を増やすと，右脚に取り付けられた球と左脚に取り付けられた球のすべての組み合わせに関して距離に関する拘束条件を考慮しなければならず，計算時間が増大しオンラインでの使用が困難となってしまう．実際には可動範囲の制限や本稿では歩行動作を生成対象としていることから干渉が発生する部位は限られているため，様々な着地位置位置列を入力して歩行動作を生成した際に脚同士の干渉が発生しない様に球の個数，配置，サイズを調整し，Fig. 6 に示したような構成を得た．HRP-2 は腿の間の空間を大きく取る片持ち構造を採用しており腿同士の干渉は発生しないこと，本稿では歩行動作を生成対象としており腿と他の部位との干渉が予測されないことから腿には幾何情報を設定していない．

提案する手法を両脚の全 12 自由度に対して適用した．遊脚の水平面内の位置に対しては厳密度可変拘束を適用するため $\boldsymbol{e}$, $\boldsymbol{S}_{waist}$, $\boldsymbol{S}_{foot}$ を以下のように定義し，支持脚期には重みが 10^5, 遊脚期の中間では 10^3 となるように $\boldsymbol{W}_e$ を設定した．

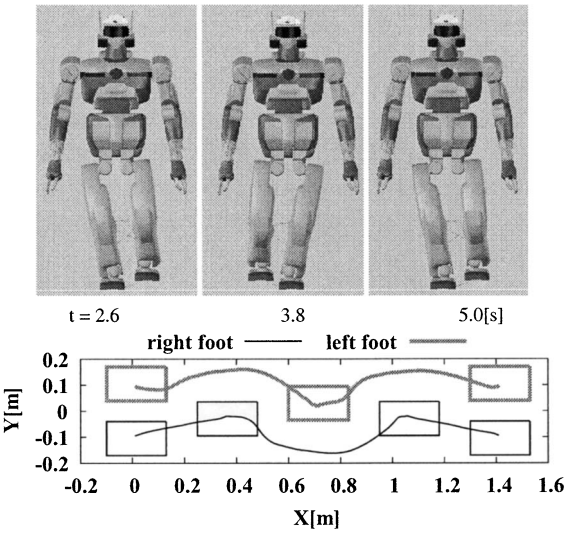

Fig. 7 Foot trajectories and postures generated using the constraint solver which considers physical constraints. The swing leg is pushed outside to avoid collision

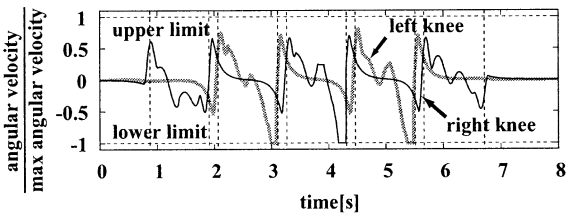

Fig. 8 Knee joint velocities of cross-legged walking. Velocities never exceed limits

$$\boldsymbol{e} = \begin{bmatrix} e_x \\ e_y \end{bmatrix},\ \boldsymbol{S}_{waist} = \boldsymbol{0},\ \boldsymbol{S}_{foot} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \\ \multicolumn{2}{c}{\boldsymbol{0}} \end{bmatrix}. \tag{11}$$

これにより最適化問題の次元は 14 となる．また拘束条件数は腰および遊脚の位置姿勢を拘束するための等式拘束条件が 12（うち厳密度可変拘束が 2），関節可動範囲，角速度を拘束するための不等式拘束条件が 24，干渉回避のための不等式拘束条件が 8（最大値．距離に応じて増減する）で，合計 44 の拘束条件下での最適な関節角速度を求めることで脚の運動生成を行った．

Fig. 7 上段に Fig. 5 上段と同時刻のロボットの姿勢を，下段に提案する手法を用いて生成された足先の軌道を示す．干渉を回避するために遊脚が支持脚の内側を回り込むような軌道が生成されていることが確認できる．

Fig. 8 に両脚膝関節の角速度の設定した最大角速度に対する比率の変化を示す．脚を大きく回り込ませる中間の 3 歩の後半で膝関節を伸展させる際に角速度が大きくなるが，設定した最大角速度に抑えられていることがわかる．また **Fig. 9** に設定した八つの干渉回避拘束のうち，一方の脚のかかとに設定した球と他方の爪先に設定した球の間に設定した拘束の距離の変化を示す．この例では式（4）における d_s, d_i, ξ をそれぞれ 0.03 [m], 0.2 [m], 1.5 [m/s] に設定したため，常に 0.03 [m] 以上の距離が

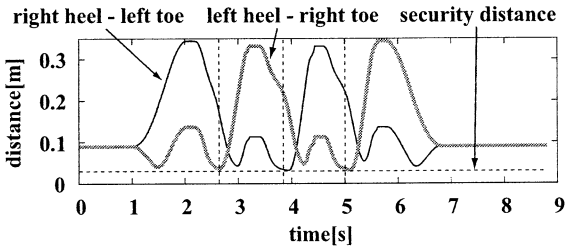

Fig. 9 Distances between feet while walking. Distances are always bigger than the security distance

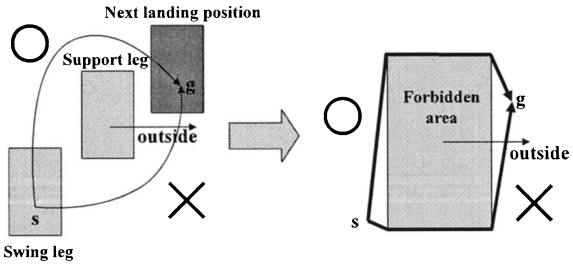

Fig. 10 Two candidates of a swing leg trajectory. The shortest path which doesn't cross with the outside vector is used to generate a swing leg trajectory

保たれていることが確認できる.

4. 遊脚軌道生成器

前章で述べた制約解決器はあくまでローカルな情報に基づくものであり，入力される拘束条件がよく設計されており，比較的小さな修正で物理的制約を満たす動作が生成可能であることが前提となる．したがって例えば前章で示した例で，着地位置の左右方向の距離をより小さくしたり，左右の脚が完全に交差するような着地位置にすると，前章で述べた手法を適用しただけでは本来支持脚の内側を移動すべき遊脚が，干渉を回避しようとした結果支持脚の外側を移動しようとして最終的には支持脚との干渉を防ぐために止められてしまい，歩行動作の生成を継続できなくなってしまうような場合が発生する．これは局所的な情報のみに基づく手法の本質的な制限である.

異なるアプローチとして，RRT [18] 等のランダムサンプリングを用いた動作計画手法を遊脚軌道の生成に用いる方法も考えられる．このような手法を用いることで脚同士の干渉が発生しない軌道を計画することは可能であるが，遊脚の軌道生成を制御周期毎に行うには計算負荷が高いという問題がある.

歩行動作生成の場合は未来の情報として次の着地位置の情報を利用することができるため，これを用いて局所解に陥りにくい遊脚軌道を生成して制約解決器に与えることで多くの場合に局所解に陥ることを防ぐことができる．ただしここでも遊脚軌道生成器は制御周期毎に軌道生成を行う必要があるため，その計算は短時間で完了するものでなければならない．そこで遊脚軌道生成器は両足先の干渉のみを考慮して脚の間に干渉が起こりにくいと思われる軌道を生成するに留め，発生する干渉は制約解決器が回避するものとする．したがってここでは腰の運動や脚の足先以外の部分を考慮せず，支持脚の位置，遊脚の現在位置と次の着地位置のみから遊脚軌道を生成する.

Fig. 10 左に示すように，遊脚の軌道には常に二つの候補が存在するが，遊脚が支持脚の外側を移動することによって脚同士の干渉が引き起こされるため，二つの経路のうち，支持脚の中心から外側方向に伸ばした半直線をまたがない経路を常に選択するものとする．足部同士の干渉を防ぐために遊脚の中心点が通過すべきでない領域は支持脚の足部の形状を表す二次元ポリゴンと遊脚の形状を表す二次元ポリゴンを原点に移動した図形の Minkowski 差によって計算されるため，これを用いて干渉が発生しない最短経路を Fig. 10 右に示すように求める.

生成された最短経路を遊脚が移動した場合，角の部分を通過する際や，次の着地位置が変更されて経路の形状が変化した場合に速度が不連続に変化する遊脚軌道が生成されてしまう．これを防ぐため，実際に遊脚に与える参照軌道は **Algorithm 1** を用いて生成する.

Algorithm 1 calcSwingLegPosition($path, t_{remain}, x, v, a$)

1: **if** $t_{remain} > t_{offset}$ **then**
2: $target \leftarrow$Divide($path, \Delta t/(t_{remain} - t_{offset})$)
3: HoffArbib($t_{offset}, target, x, v, a$)
4: **else**
5: $target \leftarrow$Divide($path$, 1.0)
6: HoffArbib($t_{remain}, target, x, v, a$)
7: **end if**

Algorithm 2 HoffArbib(t_{remain}, $goal$, x, v, a)

1: $\dot{a} \leftarrow -9a/t_{remain} - 36v/t_{remain}^2 + 60(goal - x)/t_{remain}^3$
2: $a \leftarrow a + \dot{a}\Delta t$
3: $v \leftarrow v + a\Delta t$
4: $x \leftarrow x + v\Delta t$

引数の $path$ は計算された最短経路，t_{remain} は現在動作中の一歩が着地するまでの残り時間，x，v，a は現在の遊脚の位置・速度・加速度である．関数 Divide($path$, $ratio$) は $path$ を比率 $ratio$ で内分する点の座標を返す関数，Δt は制御の時間刻みである．これによって次の着地位置が変化しなかった場合 $target$ はこの最短経路上を一定の線速度で移動し，着地時刻よりも t_{offset} だけ早く着地位置に到達する．遊脚の軌道は **Algorithm 2** に示す Hoff&Arbib による躍度最小の運動生成モデル [19] を用いて生成する．引数の x，v，a は現在の位置，速度，加速度，$goal$ は目標到達位置，t_{remain} は目標位置に到達するまでに残された時間である．これによって目標位置が制御周期毎に変更された場合でも加速度連続な遊脚軌道を生成することが可能となる.

最短経路とそれを用いて生成された遊脚軌道の例を **Fig. 11** に示す．図中でグレーで示したのが遊脚が通過すべきでない領域，その領域から右方向に出ている線分が，支持脚の中心点から支持脚に対して外側に伸びる線分である．×印が次の目標着地位置，太い実線が遊脚器道生成の元となる最短経路で破線が実際に生成された遊脚軌道である．四つの例のうち，(a)〜(c) は着地点に変更がなかった場合の軌道であり，(d) は目標着地位置を途中で変更した場合の例である．Algorithm 1 に示した

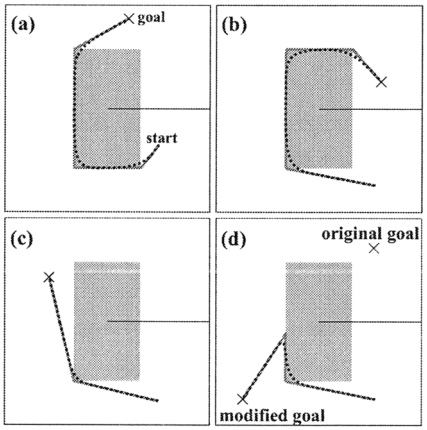

Fig. 11 Examples of swing leg trajectories. (a), (b) and (c) are cases landing positions are not changed. (d) is a case it is changed

方法で遊脚軌道を生成することによって実際に生成される遊脚の軌道は通過すべきでない領域に踏み込んだ軌道となる．しかしこの干渉は足部が同一水平面内で移動した場合に発生するものであり，実際には両足部には高低差があるため干渉が発生しない場合もありうるし，逆に遊脚軌道生成では考慮していない足部以外の部位での干渉が発生する場合もありうる．干渉が発生した場合でも，元となる遊脚軌道が足部の配置を考慮して生成されているために，歩行動作生成が継続できなくなるような深刻なものではなく，前章で提案したローカルな手法で回避可能であることが期待できる．

5. 着地位置補正

ここまでは与えられた着地位置が常に物理的制約下で実現可能であることを前提としてきた．しかし実際には着地位置生成アルゴリズムが足部の形状等を考慮していないために着地位置において足部の間に干渉が生じたり，考慮していても脚の他の部位の干渉等によって与えられた着地位置が実現不可能である場合が生じる．このような場合にも実現可能な歩行動作を実現するため，必要に応じて着地位置の変更を行うための拡張を行う．着地位置での干渉を防ぐための着地位置補正，着地位置への到達を可能とするための着地位置補正の二つの処理を各制御周期において行う．

5.1 着地位置での干渉を防ぐための着地位置補正

指定された次の着地位置において，足部同士の干渉が予測される場合には，重心軌道生成器および遊脚軌道生成器を実行する前に次の着地位置を最小の水平移動によって干渉が発生しないものに修正する．両足部の間に干渉が発生するかどうかは，両足部の形状を表す二次元ポリゴンの Minkowski 差を計算し，原点がその計算された二次元ポリゴンに含まれるかどうかで判定できる．その干渉を回避するために必要な最小並行移動は原点から Minkowski 差の境界上で原点に最も近い点へのベクトルとなるため，このベクトルを用いて着地位置を補正する．**Fig. 12** に一例を示す．この例では支持脚足部の左前と次に着地する足部の右後で干渉が発生することが予測されるため，次の着地位置を左方向に並進させることによって干渉を回避している．

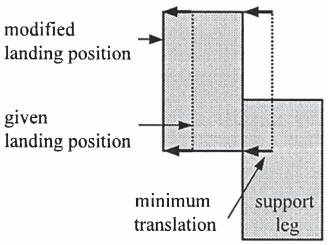

Fig. 12 Landing position modification(1): Overlap between feet is solved by the minimum translation

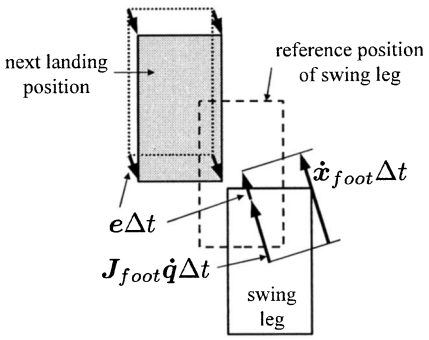

Fig. 13 Landing position modification(2): The next landing position is adjusted using an error vector $\boldsymbol{e}$

5.2 着地位置へ到達するための着地位置補正

脚同士の干渉や過大な遊脚速度の要求によって制御周期ごとに指定された位置に到達不可能である場合，厳密度可変拘束は遊脚の位置に参照軌道からの誤差を発生させることで物理的に実現可能な脚動作を生成する．この制御周期ごとに発生する誤差が累積することによって次の着地位置に到達できなくなる場合がある．このような場合に対応するため，各制御周期で厳密度可変拘束に生じた誤差の分だけ次の着地位置を修正する．この修正方法を **Fig. 13** に示す．ある制御周期に与えられた遊脚の目標移動量 $\dot{\boldsymbol{x}}_{foot}\Delta t$ に対して，最適化問題（9）を解いた結果実際の移動量は $\boldsymbol{J}_{foot}\dot{\boldsymbol{q}}\Delta t$ となり $\boldsymbol{e}\Delta t$ だけの誤差が生じる．遊脚が到達できなかったこの誤差分だけ次の着地位置を修正することで誤差の累積を防ぐ．これを繰り返すことで次の着地位置が到達不可能な位置である場合には着地位置が到達可能な範囲に引き寄せられる．

5.3 着地位置補正の例

Fig. 14 に着地位置が補正される例を示す．上段は着地位置および遊脚軌道を示す．従来の遊脚軌道生成器を用いた場合の軌道を破線で，足部の位置関係を考慮した遊脚軌道生成器を用いた場合の軌道を実線で示している．従来の軌道生成器を用いた場合，2 歩目の右足着地位置が支持脚である左足よりも左側にあるため，左足の左側（外側）を通過しようとするが，脚が交差した際の干渉を回避しようとした結果着地位置に到達できず動作生成が失敗する．これに対して足部の位置関係を考慮した遊脚軌道生成では左足の右側（内側）を通過する軌道が生成されている．着地位置に関しては，入力された着地位置を破線で，実現された着地位置を実線で示している．2 歩目，3 歩目

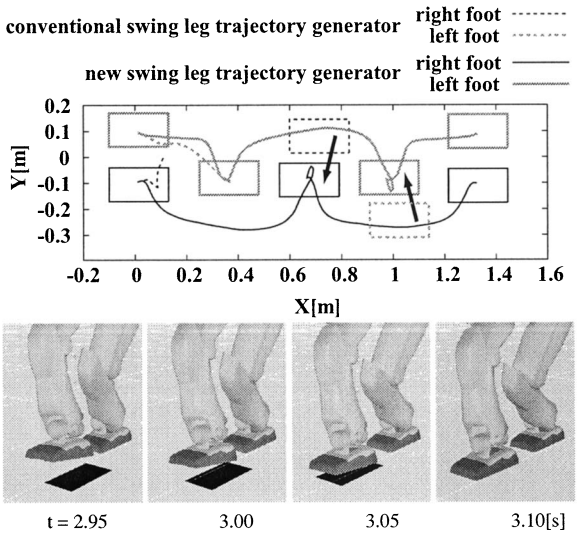

Fig. 14 Example of the next landing position modification. The second and third landing positions are modified since they are not reachable

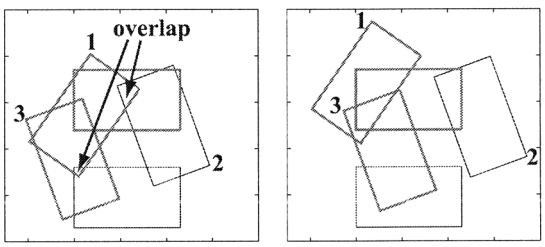

Fig. 15 Original (left) and modified (right) footprints of a wide step turning motion. Overlaps in the first and second steps are eliminated by the minimum translations

に両脚を大きく交差させるような着地位置が指定されているが，これは物理的制約を考慮した場合実現不可能な着地位置であるため，矢印で示したような補正が行われている．Fig. 14 下段は 2 歩目の着地直前のスナップショットである．床面上に描かれた矩形領域が次の着地位置であり，当初次の右脚着地位置は支持脚である左足の左前方に指定されている．生成された遊脚参照軌道に従って右足を動かしていくが，左足の膝と右足の脛の距離が小さくなるにつれ右足を左方向に移動させることが困難となるため，次の着地位置を右方向へと移動させ，最終的に着地位置は左足のほぼ前方へと移動している．

Fig. 15 に Fig. 1 に示した旋回動作時の着地位置を示す．1 歩目に左足を 55 [deg] 開き，2 歩目で右足を 55 [deg] 内股にし，3 歩目で両足を並行に戻す動作で，Fig. 1 に示したのは 2 歩目が着地したときのスナップショットである．Fig. 15 左が入力された着地位置列であり，1 歩目では踵の部分で，2 歩目では爪先の部分で両足が干渉するような着地位置列となっている．Fig. 15 右が着地位置補正を有効にした場合の着地位置で 1 歩目，2 歩目での干渉が解消されていることが確認できる．（両足の形状が接していないのは支持脚の周辺に 3 [cm] の安全マージンを取っているためである．）

Fig. 16 上段はこの旋回動作中の腰高さを，下段は膝と脛に

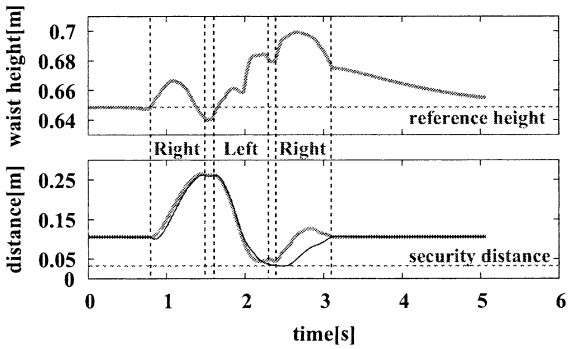

Fig. 16 Waist height and distances between knees while turning. The waist is pushed up to avoid collisions between knees

設定した干渉回避用の球の間の距離を示したものである．縦の破線は単脚支持期と両脚支持期の境界を表す．この旋回動作時には，腰の高さに対する制約に関しても厳密度可変拘束を用いるため $\boldsymbol{e}$, $\boldsymbol{S}_{waist}$, $\boldsymbol{S}_{foot}$ を以下のように設定し，e_z に対する重みは 1.0 とした．

$$\boldsymbol{e} = \begin{bmatrix} e_x \\ e_y \\ e_z \end{bmatrix},\ \boldsymbol{S}_{waist} = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 1 \\ & \mathbf{0} & \end{bmatrix},\ \boldsymbol{S}_{foot} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \\ & \mathbf{0} & \end{bmatrix}. \tag{12}$$

2 歩目で内股になる際には膝の間の距離が小さくなるため，腰の位置を上げることで膝を伸ばして膝の干渉を回避していることが分かる．干渉が発生しない 1 歩目においても腰の高さを変化させているのは，遊脚を素早く上げ下げする代わりに，支持脚も使って遊脚の高さを変化させることで最適化問題（9）の目的関数を最小化した結果である．

6. 実　　験

Fig. 5 および Fig. 15 に示した着地位置列を入力として生成された動作パターンを HRP-2 の実機に適用し，実機で実現可能な動作パターンが生成されていることを確認した．これらのパターンは $\boldsymbol{e}$ を式（12）に示したものとし，脚の位置関係を考慮した遊脚軌道生成器と着地位置補正を有効にして生成したものである．実験の際のスナップショットを **Fig. 17**，**Fig. 18** に示す．いずれの場合にも動作は正常に実行されていることが確認できる．

Fig. 19 に Fig. 17 に示した動作を生成した際の処理時間の変化を示す．動作生成を行った計算機の CPU は Pentium III 1.26 [GHz] である．HRP-2 の場合，制御周期 Δt は 5 [ms] であり，動作生成以外にもバランス制御等の処理を行う必要があるため，動作生成に使用できる時間は最大で 3 [ms] 程度である．この動作生成においては，平均処理時間が 0.3 [ms]，最長時間が 0.7 [ms] で CPU が幾分古いものであるにもかかわらず十分に短時間で動作生成ができていることが分かる．したがって現状のアルゴリズムはより多自由度なシステムへの適用や拘束条件の追加にも対応可能である．

ヒューマノイドロボットにおいてはバランス維持は必須の条

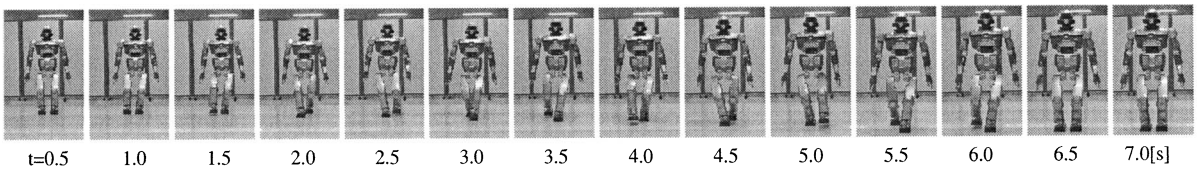

Fig. 17 Snapshots of experiments(1): cross-legged walking

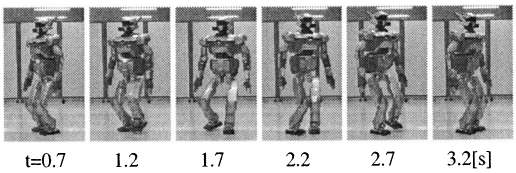

Fig. 18 Snapshots of experiments(2): wide step turning

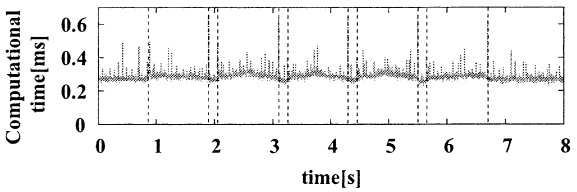

Fig. 19 Computational time to generate cross-legged walking. It is enough short to generate patterns online

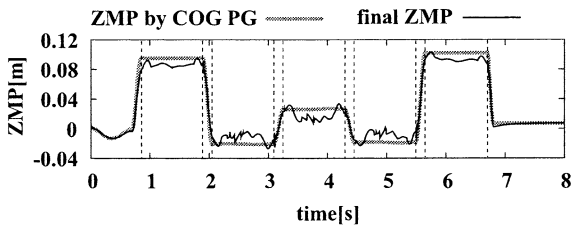

Fig. 20 ZMP trajectory generated by COG pattern generator and final ZMP trajectory. ZMP error is around 2 [cm] and enough small

件であるが，提案した手法はこれを考慮していない．これは物理的制約を満たすために動作に与える修正が比較的小さなものであり，ZMP に与える影響が小さいという仮定に基づくものである．Fig. 17 に示した動作を生成した際の，重心軌道生成器が生成した ZMP および生成された動作の ZMP の左右方向の成分を **Fig. 20** に示す．二つの軌道の差は 2 [cm] 程度であり，これには単質点での計算とマルチボディでの計算による差も含まれるため，少なくともこの動作においては ZMP に与える影響は非常に小さいと言える．

しかし Fig. 14 に示した動作の場合には，単脚支持期が終了する直前に着地位置の補正を行っており，このような着地位置の補正は重心軌道生成器の出力の段階で大きな ZMP 変動を伴なう [20]．**Fig. 21** に理想的な ZMP と重心軌道生成器によって生成された ZMP を示す．大きく着地位置を変更した 2 歩目，3 歩目で大きな ZMP 変動が起きていることが分かる．この問題に対処するためには，重心軌道生成アルゴリズムの変更，着地位置補正の変更が必要と考えられ，今後の課題の一つである．

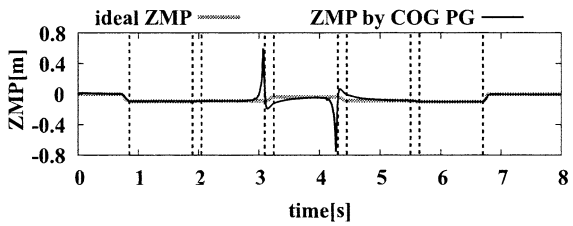

Fig. 21 ZMP trajectory generated by COG pattern generator and ideal ZMP trajectory. Big fluctuations are caused by landing position modifications just before landing

7. 結　　言

本稿では，実際のロボットハードウェアで実現可能な動作生成を行うために，物理的制約を考慮した脚動作生成手法を提案した．この手法は，動作が実現可能であるための物理的制約を関節速度に対する線形等式，不等式条件として記述し，線形 2 次計画問題を解くことで最適な関節速度を求める制約解決器とその手法が局所解に陥らないようにするための両脚の位置関係を考慮した遊脚軌道生成器からなる．また厳密度可変拘束，着地位置補正機能を導入することで実現可能な関節速度が発見される可能性を高めた．

この着地位置補正機能を用いると歩行動作生成器に入力された着地位置列が特定の座標に到達することを目的としたものである場合に目標位置には到達できなくなってしまう．この問題に対処するためには実際に実行された着地位置の情報を着地位置計画部分にフィードバックして計画を随時更新することが必要である．

また提案した手法は必ず物理的制約を満たした動作が生成されることを保証することはできない．例えば両脚支持期においては両足先が拘束されるため，物理的制約を満たすために利用できる自由度が少なく，解が存在しない場合が発生しやくすなる．このような場合にはさらに拘束を緩める範囲を広げる等の対応が必要であり，これをどのように行うかは今後の課題である．

謝　辞　本研究は JST 戦略的国際科学技術協力推進事業「実環境のオンライン情報構造化を用いたロボットの運動計画および実行に関する研究」の支援を受けてなされたものである．また提案手法の実装には QuadProg++ [21] を uBlas を用いたものに変更した uQuadProg [22] を用いた．これらのソフトウェアの開発者に感謝する．

参 考 文 献

[1] Y. Kuroki, B. Blank, T. Mikami, P. Mayeux, A. Miyamoto, R. Playter, K. Nagasaka, M. Raibert, M. Nagano and J. Yamaguchi: “Motion Creating System for A Small Biped Entertain-

ment Robot," Proc. of the IEEE/RSJ International Conference on Intelligent Robotics and Systems (IROS'03), pp.1394–1399, 2003.

[2] M. Ruchanurucks, S. Nakaoka, S. Kudoh and K. Ikeuchi: "Humanoid Robot Motion Generation with Sequential Physical Constraints," Proc. of the 2006 IEEE International Conference on Robotics & Automation, pp.2649–2654, 2006.

[3] J. Kuffner, S. Kagami, K. Nishiwaki, M. Inaba and H. Inoue: "Online Footstep Planning for Humanoid Robots," Proc. of the 2003 IEEE International Conference on Robotics & Automation, pp.932–937, 2003.

[4] N. Perrin, O. Stasse, F. Lamiraux, P. Evrard and A. Kheddar: "On the Problem of Online Footsteps Correction for Humanoid Robots," 第 27 回日本ロボット学会学術講演会予稿集 DVD-ROM, pp.AC3O1–04, 2009.

[5] P. Baerlocher and R. Boulic: "An Inverse Kinematics Architecture Enforcing an Arbitrary Number of Strict Priority Levels," The Visual Computer: International Journal of Computer Graphics, vol.20, no.6, pp.402–417, 2004.

[6] H. Sugiura, M. Gienger, H. Janssen and C. Goerick: "Real-Time Collision Avoidance with Whole Body Motion Control for Humanoid Robots," Proc. of International Conference on Intelligent Robots and Systems, pp.2053–2058, 2007.

[7] O. Stasse, A. Escande, N. Mansard, S. Miossec, P. Evrard and A. Kheddar: "Real-Time (Self)-Collision Avoidance Task on a HRP-2 Humanoid Robot," Proc. of the 2008 IEEE International Conference on Robotics & Automation, pp.3200–3205, 2008.

[8] J. Kuffner, K. Nishiwaki, S. Kagami, Y. Kuniyoshi, M. Inaba and H. Inoue: "Self-Collision Detection and Prevention for Humanoid Robots," Proc. of the 2002 IEEE International Conference on Robotics & Automation, pp.2265–2270, 2002.

[9] K. Nishiwaki, S. Kagami, Y. Kuniyoshi, M. Inaba and H. Inoue: "Online Generation of Humanoid Walking Motion based on a Fast Generation Method of Motion Pattern that Follows Desired ZMP," Proc. of the IEEE/RSJ International Conference on Intelligent Robotics and Systems (IROS'02), pp.2684–2689, 2002.

[10] B. Mirtich: "VClip: Fast and robust polyhedral collision detection," ACM Transactions on Graphics, vol.17, no.3, pp.177–208, 1998.

[11] M. Vukobratović and B. Borovac: "Zero-moment point—thirty five years of its life," International Journal of Humanoid Robotics, vol.1, no.1, pp.157–173, 2004.

[12] M. Morisawa, K. Harada, S. Kajita, K. Kaneko, F. Kanehiro, K. Fujiwara, S. Nakaoka and H. Hirukawa: "A Biped Pattern Generation Allowing Immediate Modification of Foot Placement in Real-time," Proc. of the IEEE-RAS International Conference on Humanoid Robots, pp.581–586, 2006.

[13] F. Kanehiro, W. Suleiman, K. Miura, M. Morisawa and E. Yoshida: "Feasible Pattern Generation Method for Humanoid Robots," Proc. of the IEEE-RAS International Conference on Humanoid Robots, pp.542–548, 2009.

[14] B. Faverjon and P. Tournassoud: "A Local Based Approach for Path Planning of Manipulators With a High Number of Degrees of Freedom," Proc. of IEEE International Conference on Robotics and Automation, pp.1152–1159, 1987.

[15] 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.

[16] M. Benallegue, A. Escande, S. Miossec and A. Kheddar: "Fast $\mathcal{C}^1$ Proximity Queries using Support Mapping of Sphere-Torus-Patches Bounding Volumes," IEEE International Conference on Robotics and Automation (ICRA), pp.483–488, Kobe, Japan, 12-17 May 2009.

[17] 金広文男，吉田英一，Lamiraux Florent, Kanoun Oussama, Laumond Jean-Paul："任意の多面体間に適用可能な干渉回避運動生成法"，日本ロボット学会誌，vol.27, no.1, pp.63–70, 2009.

[18] S.M. LaValle: Rapidly-exploring random trees: A new tool for path planning, Technical Report 98-11, Computer Science Dept., Iowa State University, 1998.

[19] B. Hoff and M.A. Arbib: "Models of Trajectory Formation and Temporal Interaction of Reach and Grasp," Journal of Motor Behavior, vol.25, no.3, pp.175–192, 1993.

[20] M. Morisawa, K. Harada, S. Kajita, K. Kaneko, J. Sola, E. Yoshida, N. Mansard, K. Yokoi and J.P. Laumond: "Reactive Stepping to Prevent Falling for Humanoids," Proc. of the IEEE-RAS International Conference on Humanoid Robots, pp.528–534, 2009.

[21] D. Goldfarb and A. Idnani: "A numerically stable dual method for solving strictly convex quadratic programs," Mathematical Programming, vol.27, pp.1–33, 1983.

[22] uQuadProg, http://www.lis.deis.unical.it/~furfaro/uQuadProg/uQuadProg.php.

金広文男（Fumio Kanehiro）

1999 年東京大学大学院工学系研究科情報工学専攻博士課程修了．博士（工学）．1998 年より日本学術振興会特別研究員．2000 年電子技術総合研究所入所．現在産業技術総合研究所知能システム研究部門主任研究員．2007 年より 1 年間 LAAS-CNRS 客員研究員．ヒューマノイドロボットのシステム構成法，全身行動制御に興味がある．IEEE の会員．

（日本ロボット学会正会員）

Suleiman Wael

Wael Suleiman received his Master and PhD degrees in Automatic Control from Paul Sabatier University in Toulouse, France, in 2004 and 2008 respectively. He is now postdoctoral researcher of the Japan Society for the Promotion of Science (JSPS) at CNRS-AIST JRL, UMI3218/CRT, at National Institute of Advanced Industrial Science and Technology (AIST), Tsukuba, Japan. His research interests include nonlinear system identification and control, numerical optimization and humanoid robots.

吉田英一（Eiichi Yoshida）

1996 年東京大学大学院工学系研究科博士課程修了，同年通産省工業技術院機械技術研究所入所，2001 年 4 月より独立行政法人産業技術総合研究所・主任研究員，博士（工学）．2004〜2009 年フランス LAAS-CNRS にて日仏ロボット工学共同研究ラボラトリー（JRL-France）共同ディレクター．現在 AIST-CNRS ロボット工学連携研究体長として，フランス CNRS との共同研究に従事．作業・動作計画，モジュール型ロボット，人間型ロボットの制御に興味を持つ．日本機械学会，計測自動制御学会，IEEE などの会員．

（日本ロボット学会正会員）

森澤光晴（Mitsuharu Morisawa）

2004 年慶應義塾大学大学院理工学研究科総合デザイン工学専攻修士博士課程修了．博士（工学）．同年 4 月より独立行政法人産業技術総合研究所知能システム研究部門研究員．2009 年 4 月より 1 年間，仏国 LAAS-CNRS 客員研究員．パラレルメカニズム，モーションコントロール，ヒューマノイドロボットなどの研究に従事．

（日本ロボット学会正会員）

金子健二（Kenji Kaneko）

1990 年慶應義塾大学大学院理工学研究科電気工学専攻修士課程修了．同年工業技術院機械技術研究所に入所．1995 年 9 月より半年間米国カーネギーメロン大学客員研究員，1999 年 9 月より 1 年間仏国 CNRS パリロボット研究所客員研究員．2001 年より組織改変に伴い独立行政法人産業技術総合研究所主任研究員，現在にいたる．博士（工学）．モーションコントロール，マイクロマシン，遠隔制御，ヒューマノイドロボット等の研究に従事．日本機械学会，電気学会の各会員．

（日本ロボット学会正会員）