バランス制御と共存可能な ヒューマノイドの障害物回避アルゴリズム

藤原 清 司* 金 広 文 男* 比留川 博 久*

A Collision Avoidance Algorithm of Humanoid Robots that can be used together with a Balance Controller

Kiyoshi Fujiwara*, Fumio Kanehiro* and Hirohisa Hirukawa*

A collision free motion of a humanoid robot is perturbed when a balance controller is applied to make the robot stable, then the motion may cause collisions. This paper presents a collision avoidance algorithm of humanoid robots that can be used together with a balance controller. The proposed algorithm cancels the perturbations at shoulder joints, and the sweeping volume of an arm can be kept same even when a balance controller is applied to the robot. The prerequisite of the algorithm is that three axes of shoulder joints should pass through a point.

Key Words: Humanoid Robot, Motion Planning, Balance Control

1. は じ め に

ヒューマノイドロボットの障害物回避動作計画問題は、二つの点で通常のマニピュレータの障害物回避問題と性質が異なる。第1点は、マニピュレータのように一端が環境あるいは安定で十分重い車台に固定されていないことから、障害物を回避するという目的だけでなく、外乱に抗しつつ姿勢を安定とするという目的を同時に満足しなければならない点である。第2点は、ヒューマノイドロボットは人間を模した構造と多くの自由度を持つため、環境だけではなく自分自身との衝突が起こりやすい点である。多くのヒューマノイドロボットにおいて、腕の掃引体積の何割かは、自己干渉を発生させる。また、極端な姿勢においては、腕と脚も衝突する場合がある。

Kuffnerらは、この問題を2段階に分けて扱う方法を提案している[1]。これは、まず障害物を回避する関節軌道の集合を求め、この軌道を全身姿勢が安定となるように変換することにより、二つの目的を満足させる方法である。この方法は、シミュレーションおよび実験によりその有効性が確認されており、state of the artの技術であると言える。しかしながら、第2段階の変換を軌道に施すことによりロボットの掃引体積は変化するので、障害物を回避することはもはや保証されない。Kuffnerらは、変換後の動作に対して干渉チェックをし、干渉する場合は第1段階からやり直す方法を採用している。この方法の場合、実時間で姿勢を変化させて外乱に応じ全身のバランスを取るフィードバック

制御系との共存は困難であると考えられる。

本稿では、ヒューマノイドロボットの機能的な特徴に着目して、これら二つの問題を分離する方法を提案する。腕による作業を想定したヒューマノイドロボットにおいては、通常、人間の肩に相当する部分に三つの回転自由度が用意されている。ここで注目したいのは、多くの場合、それら三つの回転軸は1点において交差するという点である。我々はこの点に着目し、バランス制御によって変動した肩の並進誤差を脚の自由度によって打ち消し、さらに回転誤差を肩3軸の回転によって補うという補償方法を提案する。

本研究が目的とする作業内容は、ヒューマノイドが直立したまま片手で行う作業である。具体的にはプラント内のバルブ操作や、スイッチボックス内のスイッチ操作、棚の中の物品の把握などが該当する。この目的から、ここで提案する方法は、作業を行う片腕が作業環境およびロボット自身に干渉しない動作の計画を行う。また、脚および胴体部分と、環境との衝突を考慮する必要がない程度に離れた位置に立つことができるものとする。これは、作業に用いる腕と異なり、胴体や脚は狭隙部への挿入や対象との接触を行う必要がないためである。このとき、事前の情報やセンサ情報による認識によって環境モデルおよびロボットとの位置関係は得られているものと仮定する。

以下、第2章ではバランス制御と計画された動作の実現を両立させる方法の詳細について述べる。第3章ではこれを前提とした障害物回避動作計画法について述べる。第4章では本手法の実装方法とシミュレーション例を示す。最後に第5章において結論を述べる。

原稿受付 2002年1月23日

*独立行政法人産業技術総合研究所

*AIST

2. バランス制御と障害物回避の分離方法

2.1 ヒューマノイドの自由度配置

前述したように、本論文ではヒューマノイドが片腕で行う作業を対象とし、その際に環境と胴体および脚部が干渉する可能性は考慮する必要がないものとする。こうした前提のもとで、本論文で提案する手法の適用条件は、次の通りである。

- (1) 肩3軸の回転軸が1点で交わる自由度構成を持つヒューマノイドであること。

現在までに開発されているヒューマノイドロボットのほとんどはこの適用条件を満たしており、非常に緩やかな前提条件であると考えられる。

この前提条件を満たすヒューマノイドの例として、HRP (Humanoid Robotics Project of METI) [2] [3] において用いられている HRP-1 を考える。HRP-1 の自由度配置を Fig. 1 に示す。

HRP-1 のそれぞれの脚は、股関節部に3自由度 (LEG[1], LEG[2], LEG[3]), 膝に1自由度 (LEG[4]), 足首部に2自由度 (LEG[5], LEG[6]) の6自由度を備える。腕は、肩部に3自由度 (ARM[1], ARM[2], ARM[3]), 肘部に1自由度 (ARM[4]), 手首部に3自由度 (ARM[5], ARM[6], ARM[7]) の7自由度からなる (Table 1)。肩の3自由度 (ARM[1], ARM[2], ARM[3]) は、共有される関節原点において交差する三つの回転軸からな

Fig. 1 Configuration of humanoid robot HRP-1

Table 1 Joint locations of humanoid robot HRP-1

LEG[1], LEG[2], LEG[3]	hip
LEG[4]	knee
LEG[5], LEG[6]	ankle
ARM[1], ARM[2], ARM[3]	shoulder
ARM[4]	elbow
ARM[5], ARM[6], ARM[7]	wrist

るため、球関節と同様に捉えることが可能である。このことは手首や股関節においても同様である。腕7自由度、脚6自由度であるため、合計すると全身の自由度は26自由度になる。

このような自由度配置を持つヒューマノイドに対して、障害物回避動作計画モードとバランス制御適用モードで異なる自由度の利用法を用いるが、両モードに共通するのは、

- (2) 作業中足位置は変更しないこととし、脚関節は6自由度のリンク機構とみなせるものとする。
 (3) 片腕の障害物回避のみを考慮し、もう片方の腕はその間動かさないものとする。
 (4) 動かす方の腕の手首3関節も、障害物回避動作中は固定するものとする。

という点である。(3)において片腕のみを対象としているのは、両腕に対して本手法を適用することが計算量の点から容易ではないためである。(4)で手首3関節を固定することとしているが、一般に手首部の掃引体積は小さいので、これを手首の回転中心点を中心とした球で近似することにすれば、任意の手首関節角に対して障害物と干渉しない動作を計画することも可能である。

障害物回避動作計画モードでは、これらに加えて
 (5) 脚の2自由度、肩3関節、肘関節の計6自由度を用いて障害物回避動作計画を行う。

ものとする。後述するように、足首の2関節をバランス制御に利用するのが、力学的には望ましい選択肢の一つである。

この場合、動作計画に利用する2自由度は残りの4自由度から選択することになるが、一般に股関節のピッチ軸、膝関節は可動範囲が広いので、これらの2軸を用いるのが一つの合理的な選択である。例えば、HRP-1の場合は、股関節のピッチ軸は140度以上、膝関節は90度以上可動するのに対して、股関節の残りの関節は45度以下の範囲しか動かない。肘関節も一般に可動範囲が広く、これを選択するのも合理的な選択であると言える。

この「可動範囲の広い関節を選択する」という評価基準の合理性については、以下の理由から妥当であると考えている。一般に、各関節の可動範囲が大きくなると手先の掃引体積は大きくなる。例えば、1関節ロボットだと一つしかない関節が広い範囲を動くと手先の描く曲線は長くなる。2関節ロボットだと、固定部に近い関節を動かしたときに二つめの関節が描く曲線の各点に対して、二つめの関節を動かしたときに描く曲線の和集合が手先の掃引面になるが、各関節が動く範囲が広いときにこの面の面積が広くなるのは容易に想像可能である。関節数が多いとき、手先の正確な掃引体積を求めることは計算量の観点から容易ではないが、例えば他の関節の可動範囲を固定して、一つの関節の可動範囲を広げれば、全体の掃引体積が増加することは想像に難くないと思われる。

掃引体積が大きくなると、取りうる手先軌道の選択肢が増える。障害物回避動作計画を行う際に利用する関節数は6なので手先の位置姿勢を定めるために冗長自由度はなく、一つの手先軌道に対応する関節軌道の数はせいぜい定数倍しかない。したがって、取りうる関節軌道の選択肢が増える。障害物回避軌道の計画は、取りうる関節軌道の中から障害物を回避する軌道を

関節空間の探索により発見することなので、関節軌道の選択肢が増えれば障害物回避軌道を発見する確率も高くなる。以上の主張を定量的に評価するためには、掃引体積を求める必要があるが、6自由度の系に対してこれを行うのは計算量の点から困難である。したがって、以上の議論は定性的ではあるが、十分合理的な基準であると考えられる。

バランス制御モードでは、

- (6) 脚関節 6 自由度のうち 2 自由度をバランス制御に利用し、残りのうち 3 関節を肩関節の回転中心の位置の制御に利用する。

- (7) 肩 3 関節は、腕リンク基部の姿勢の制御に利用する。

このうち、(6) では冗長自由度を利用するなら、3 関節の代わりに 4 関節利用することも可能である。

以上の述べた自由度の利用法を前提として、ヒューマノイドロボットの障害物回避動作を以下のようにして実現する。まず、障害物回避動作計画モードでは、脚の 2 自由度、肩 3 関節、肘関節の計 6 自由度を用いて、これら 6 自由度の初期状態から目標状態へ至る障害物回避軌道を計画する。続いて、計画された軌道をバランス制御モードに入力する。バランス制御モードでは、全身の動きに応じてバランスを取るため、障害物回避動作計画段階では固定されていた足首の 2 関節が揺動する。この結果、肩に 3 関節の回転軸の交点にある腕リンク基部の位置および姿勢が計画された軌道からずれる。

この計画軌道からのずれのうち、位置については (6) で述べたように残りの脚関節の 3 自由度以上を用いて計画された位置に戻るよう修正を行い、姿勢については (7) で述べたように肩 3 関節を用いて修正を行う。ここで注意して欲しいのは、腕リンクの掃引体積が動作計画段階のものから変化しない点である。これは、腕リンクの基部の位置が計画段階と同じであるためである。肩 3 関節の関節空間における軌道は計画時から変化しているが、腕リンク基部の姿勢は計画時と同じになる。この点が本手法の特徴である。これは、前提条件である (1) の、肩 3 関節の回転軸が 1 点で交わっていることによる。この点について、次節で詳しく説明する。

2.2 バランス制御系との共存方法

前述したようにバランス制御には脚 6 自由度のうち 2 自由度を利用するが、その選択には任意性がある。ここでは、一般性を失うことなく、足首 2 自由度をバランス制御に用いることとし、以下の議論を進める。足首 2 自由度を用いたバランス制御系の実現例としては、梶田らの手法がある [4]。これは足首トルクによって接地モーメントを高精度に制御することで、バランス制御を行うものである。着地点中心を基準としたロボット全体の重心位置を $r_G \equiv [G_x, G_y, G_z]$ 、全角運動量を $L \equiv [L_x, L_y, L_z]$ とした場合、接地モーメント $t \equiv [\tau_x, \tau_y, \tau_z]$ との関係は次式で与えられる。

$$t = \frac{d}{dt} L + M r_G \times G \quad (1)$$

ただし、 M はロボットの全質量、 G は重力加速度ベクトルである。ここで佇立状態を維持するための制御目標を $L_x = L_y = 0$ および $r_{G_x} = r_{G_y} = 0$ とすると、最も単純なフィードバック則として次式が得られる。

$$\begin{cases} \tau_x^d = -k_{px} L_x - k_{vy} r_{G_y} \\ \tau_y^d = -k_{py} L_y - k_{vx} r_{G_x} \end{cases} \quad (2)$$

k_{**} は適当なフィードバックゲイン、 τ_x^d は x 軸回りの目標トルク、 τ_y^d は y 軸回りの目標トルクである。

このバランス制御はロボットの姿勢変化に対して短い時間スパンで行われるため、手足の加速運動による動的効果に対してはバランスを維持することが可能である。

梶田らの手法において、足首 2 自由度が選択されている理由は、制御の対象であるヒューマノイドロボットが足部にコンプライアンス要素を持つ点が、足首関節のトルク制御に適しているためである。また、接地モーメントの作用点に最も近いアクチュエータであるためにモーメントの制御に適しているためでもある。

前節の前提 (2) の実現方法として、両脚関節を平行的に動かすことにすると、両脚の足先リンクに固定した座標系の原点を床面に正射影した点の midpoint を原点として、LEG[1]-LEG[6] のそれぞれの関節に対応する座標変換を $A_1, \dots, A_6$ とし、障害物回避動作を行う腕の肩関節座標の位置と姿勢をそれぞれ T_6, R_6 とすると、

$$T_6 R_6 = A_1 A_2 A_3 A_4 A_5 A_6 \quad (3)$$

という関係が成立する。ただし、 T_6 は同時変換行列の並進成分のみを、 R_6 は回転行列成分のみを表すものとする。以下、 T, R は同様の変換を表すものとする。ここで、バランス制御の結果 LEG[1] 関節および LEG[2] 関節が微小動作し、対応する座標変換が A'_1, A'_2 になったとすると、肩関節座標の位置と姿勢も変化し、

$$T'_6 R'_6 = A'_1 A'_2 A_3 A_4 A_5 A_6 \quad (4)$$

となる。このとき、 $A_3, \dots, A_6$ を $A'_3, \dots, A'_6$ に変化させて、 T'_6 を T_6 に戻すことを考える。これは、運動学方程式

$$T_6 R''_6 = A'_1 A'_2 A'_3 A'_4 A'_5 A'_6 \quad (5)$$

を満たす解があるかどうかという問題である。ここに、 R''_6 は任意の回転行列とする。式 (5) を変形すると、

$$(A'_1 A'_2)^{-1} T_6 R''_6 = A'_3 A'_4 A'_5 A'_6 \quad (6)$$

となり、 $(A'_1 A'_2)^{-1} T_6 = \hat{T} \hat{R}$ とおくと、

$$\hat{T} \hat{R} R''_6 = A'_3 A'_4 A'_5 A'_6 \quad (7)$$

とかける。さらに、 $\hat{R} R''_6 = R$ とおくと、

$$\hat{T} R = A'_3 A'_4 A'_5 A'_6 \quad (8)$$

となる。 $A'_3, \dots, A'_6$ には独立変数が四つあるので、一般に、並進変換を $\hat{T}$ とする逆運動学の解が存在する。したがって、(5) 式を満たす $A'_3, \dots, A'_6$ が存在する。このとき、障害物回避動作を行う腕の肩 3 関節 ARM[1], ARM[2], ARM[3] による座標変換を A_7, A_8, A_9 とおくと、前節で仮定した適用条件 (1) により、これら三つの変換の合成変換は一つの回転変換 R_s で表すことができる。ただし、説明を簡単にするため、肩関節原

点から肘関節への固定並進変換はここでは考えないものとする。 $R_s = A_7 A_8 A_9$ を式 (3) の右から乗じると、

$$T_6 R_6 R_s = A_1 A_2 A_3 A_4 A_5 A_6 A_7 A_8 A_9 \quad (9)$$

と書ける。ここで、一般に、球面変換の逆運動学解を求めることにより、 $R_6'^{-1} R_6 R_s = A_7' A_8' A_9'$ となるように $A_7' A_8' A_9'$ を定めることができるので、これを式 (5) の右から乗じると、

$$T_6 R_6' R_6'^{-1} R_6 R_s = A_1' A_2' A_3' A_4' A_5' A_6' A_7' A_8' A_9' \quad (10)$$

$$T_6 R_6 R_s = A_1' A_2' A_3' A_4' A_5' A_6' A_7' A_8' A_9' \quad (11)$$

となるので、障害物回避動作を行う腕の肩関節座標の位置と姿勢は、バランス制御による揺動が加わる前の状態に戻すことができる。足首の角度が変化しても肩から先の位置および姿勢が変動しないよう保持されている様子を図に示す (Fig. 2)。

以上述べた方法を適用することにより、障害物回避動作を行う腕の掃引体積は、バランス制御を適用した後も動作計画時と同一になる。Kuffnerらの方式では、障害物回避動作軌道を全身姿勢が安定となるように変換するので、変換後は掃引体積が変化し、障害物を回避することは保証されない。このため、変換後に干渉が発見された場合には動作計画をやり直す必要がある。本手法は、一般的な肩関節の構造を前提とすることにより、このようなやり直しの必要性を除去した点に特徴がある。

ヒューマノイドロボットは冗長自由度系なので、以上述べたような機構的な特徴に着目する方法ではなく、一般の冗長自由度系の動作計画手法が適用できるとも考えられる [5]~[9]。しかしながら、一般に冗長自由度系の動作計画手法は目標到達と障害物回避という二つの目標を達成するため、何らかの評価関数に従って各時点においてすべての自由度に運動を割り振り、後戻りなしの探索を行って動作を計画する。この後戻りなしの探索は非常に限定された解しか発見できない。そこで、Random Roadmap 法等の後戻りありの探索組み合わせる方法も考えら

れるが、前述したようにすべての自由度に運動が割り振られるので、すべての自由度からなる関節空間を探索空間とする必要がある。ところが、ヒューマノイドの場合は自由度数が多いので、これは計算量の観点から非現実的となる。本稿で提案する方法では、限定された関節空間において探索を行うので、計算量の観点から遥かに有利である。

3. 動作計画アルゴリズム

ここでは、前章で詳述した方法を前提として、ヒューマノイドロボットの障害物回避動作計画を行う方法を示す。作業環境およびヒューマノイドロボット自身の幾何モデルは既知であると仮定する。また、ヒューマノイドロボットの関節角および作業環境における位置・姿勢も既知であると仮定する。

ここで用いる経路計画手法は Random Roadmap 法 [10] と呼ばれる手法である。この手法は状態空間中にランダムに大量の候補点を生成し (Fig. 3)、これらノード群を結合するアークをはりめぐらせ、得られた経路の有効性を判定することを繰り返すものである。これによって、初期状態から目標状態へ遷移する有効な経路を求める (Fig. 4)。この手法は、生成するノードの数を無限に増やせば、解であるパスが存在するときに必ずこれを発見することが保証されている、すなわち、確率的完全なアルゴリズムである。障害物回避動作計画問題の複雑度は、PSPACE 困難であることが理論的に証明されており [11]、解が存在するときにこれを発見することが保証されている完全なアルゴリズムの複雑度は自由度に対して指数関数で増える [12]。このため、確率的完全なアルゴリズムの研究が盛んに行われており、現在のところ最も有望な方法である。本稿ではこの Random Roadmap 法を採用する。我々は Kavraki らのアルゴリズムを以下の点について修正した。

- アークの有効性を判定する際にはアークを再帰的に分割し、個々の分割点において有効性の判定を行う
- パス全体で、それぞれのアークを分割する細かさである粒

Fig. 2 Cancellation of the perturbation by the ankle joints

Fig. 3 Random roadmap method

Fig. 4 Random roadmap method

度が荒い段階で有効性を判定し、パス全体が有効であれば粒度を徐々に細かく設定して全体の判定を繰り返す

- 判定結果はその粒度とともに記憶し、同じアークの同じ粒度について判定を繰り返さない

ここではアークの有効性を判定するために、アークを細かく分割した個々の分割点について有効性判定を実施し、すべての分割点が無効であればそのアークを有効と確定する。しかし、ロボットと環境の干渉が発生する点は連続している可能性が高いので、初めから細かい分割を行うことは非効率である。また、パスの一部分についてのみ細かい粒度で判定することも効率的ではない。上記の修正により、パス全体を通しての環境とロボットとの干渉が早期に検出される可能性が向上する。また同じ判定を重複して繰り返すことがないため、処理の高速化が期待できる。これらの修正を含んだ最終的なアルゴリズムの詳細を以下に示す。

- (1) コンフィグレーション空間 C 中に M_1 点のノード群 $N = \{n_1, n_2, \dots, n_{M_1}\} \in C$ をランダムに生成する。ここに、 C は障害物回避計画に用いる関節座標から構成される空間である。本手法では脚2関節、肩3関節、肘関節の合計6関節座標が用いられる。個々のノード n_i の一部に対して後に有効性判定を施すが、初期化時にはその判定結果 $\lambda_i^n = \text{unknown}$ としておく。ここで、個々の関節座標は関節の可動範囲より小さい範囲をとるものとする。
- (2) N からランダムに M_2 個の組み合わせを選択し、各組についてその間を接続するアークの集合 $P = \{p_1, p_2, \dots, p_{M_2}\}$ を生成する。アークは、コンフィグレーション空間においてノード間を結ぶ線分とする。次に、各アークに近傍性判定 $\Delta(p)$ の結果を記憶する。 $\Delta(p)$ は、アーク両端のノード間の距離 d がある閾値より小さければ *near*、そうでなければ *far* を返す。ノード間の距離 d は、二つのコンフィグレーションにおける手先位置間のユークリッド距離により定義する。近傍判定結果は、 δ_j^p に記録する。また、各アークについても有効性判定は初期化時には行わず、それぞれ

のアークの有効性判定記録 $\lambda_j^p = \text{unknown}$ と初期化しておく。

- (3) 初期姿勢と目標姿勢のそれぞれに対応するノード $n_{start} \in C, n_{goal} \in C$ を生成し、 N に追加する。これらスタートとゴールとなるノードのそれぞれについて有効性判定 $\Lambda(n)$ を行う。ここに、有効性判定とは、ロボットがコンフィグレーション n にあるとき、各関節の可動範囲の制約を満たし、自己干渉せず、かつ、作業環境と干渉しないときには *valid*、それ以外の場合は *invalid* を返す判定である。 $\Lambda(n_{start})$ または $\Lambda(n_{goal})$ が *invalid* ならば解は存在しないので、障害物回避軌道は存在しないと判定して、アルゴリズムを終了する。
- (4) n_{start}, n_{goal} のそれぞれについて、 N からランダムに M_3 点ずつ選択したノード n のそれぞれに接続するアークを生成し、 P に追加する。この際生成されたアーク p_j に関する有効性判定記録 λ_j^p は初期値 *unknown* が与えられる。また、各アーク p_j に近傍判定 $\Delta(p_j)$ を実施し、その結果を δ_j^p に記録する。
- (5) λ_i^n の値が *valid* または *unknown* である $n_i \in N$ の集合をノードとし、 λ_j^p の値が *valid* または *unknown* であり、かつ、 δ_j^p の値が *near* である $p_j \in P$ の集合をアークとする無向グラフ上で、 n_{start} から n_{goal} へ至るパスを探索する。
- (6) n_{start} から n_{goal} へ至るパスが見つからない場合は、 M_4 の確率でノードを1点とそこから既存のノードに連なるアーク2本をランダムに追加する。さらに任意のノード間にランダムに、 M_5 本のアークを追加する。その後、(5) へ戻って再度経路探索を行う。 L_{max} 回本ステップを繰り返してもパスが見つからなかった場合は、解が発見されなかったとしてアルゴリズムを終了する。
- (7) n_{start} から n_{goal} へ至るパスが見つかった場合は、このパス上のすべてのノード n_i について、有効性判定 $\Lambda(n_i)$ を行い、判定結果を λ_i^n に記録する。判定結果が一つでも *invalid* になった場合は、ステップ (5) へ戻る。
- (8) パス上のそれぞれのアーク p_j について、線分を2分割して中点 m について有効性判定 $\Lambda(m)$ を行う。2分割を R 回再帰的に繰り返して、すべての判定が *valid* であったときは $\lambda_j^p = \text{valid}$ とし、そうでないときは $\lambda_j^p = \text{invalid}$ とする。
- (9) パス上で一つでも $\lambda_j^p = \text{invalid}$ となったときはステップ (5) に戻る。
- (10) 発見されたパスに短縮化処理を施したものを出力し、アルゴリズムを終了する。ここに、短縮化処理とは、発見されたパス上の近傍ノード間の中間点にノードを追加し、パス上のノードおよび追加されたノード間を線分で結ぶアークを新たに生成したのち、それぞれのアークに対してステップ (8) の判定を行うものである。すべての判定が *valid* なアークを通るようにパスを修正することで、冗長な経路を短縮することが可能となる。また、ここで生成された経路はバランス制御が可能な範囲内に収まっていることが保証されない。この安定保持限界は、シミュレーションもしくは実機での作業実行の際に、関節角が可動範囲の限界に接

近することで安全に検出可能である。この場合、該当するアーク p_j に関する有効性判定を $\lambda_j^p = \text{invalid}$ としてアルゴリズムのステップ (5) から再度探索を実施することで、検出済みの安定保持限界を回避した他の経路を求めなおすことが一つの対処方法である。

本アルゴリズムにおいて、いくつかの基数 $M_1, \dots, M_5, L_{max}$ をどう選択するかにより、アルゴリズムの総計算時間と解を発見する確率が変化する。一般に、総計算時間を多くとることで解を発見する確率はあがる。許容される計算時間にしたがって、適切な組み合わせを設計する必要がある。また、アークの長さの最大値を決める $\Delta(p)$ の閾値と、アークの分割回数 R を適切に設定することにより、干渉チェックの粒度を決定することができる。

4. 実装とシミュレーション結果

4.1 OpenHRP を用いた実装

本アルゴリズムは、HRP (Humanoid Robotics Project) において開発されたヒューマノイドロボットソフトウェア総合開発環境 OpenHRP [13] 上に実装を行った。OpenHRP の構成を Fig. 5 に示す。OpenHRP は、リンク機構の順動力学計算を行う Dynamics Simulator, 干渉チェックを行う Collision Checker, VRML97 で記述されたロボットおよび作業環境のモデルを読み込む Model Parser, 三次元グラフィックスによりロボットの動作を表示する Online Viewer, 制御系である Controller 等から構成され、CORBA (Common Object Request Broker Architecture) [14] を用いて相互接続されている。この OpenHRP 環境に動作計画のためのサーバを追加することで、提案した経路計画機能を実装した。動作計画のために用いるのは、Collision Checker と Model Parser で、Online Viewer を視覚的な結果の確認のために利用している。このうち、Collision Checker は OBB (Oriented Bounding Box) を用いた衝突判定法を実装した Rapid [15] を用いている。ここではヒューマノイドの右腕と胴体、脚、左腕、そして全身と環境についての衝突を判定している。

動作計画サーバは他のサーバと同様に ORB を経由して OpenHRP 環境と結合する。これは抽象層、評価層、実存層の三つの層からなる (Fig. 6)。抽象層では、各座標軸が 0 から

1 までの実数に正規化された探索空間で、ランダムなノードの生成、探索グラフのトポロジーの管理、グラフ上での探索等の処理を行う。評価層では、正規化された座標からロボットの関節角への変換、基数 $M_1, \dots, M_5, L_{max}$ の設定、 $\Delta(p)$ の閾値、アークの分割数 R の選択を行う。実存層では、ロボットの順運動学計算、干渉チェック等の処理を、OpenHRP の各サーバの機能を用いて実行している。

4.2 シミュレーション結果

本手法を用いたシミュレーション例を次に示す。これは腕をかけた初期姿勢から壁の穴に腕を差し込んだ目標姿勢に至る経路を探索したものである。初期姿勢と目標姿勢を Fig. 7 に示す。このとき、 $M_1 = 10, M_2 = 10, M_3 = 10, M_4 = 0.01, M_5 = 10, R = 7$ とした。 M_1, M_2 をともに 10 としたのは、後述する case3 のように容易な問題も対象となっているため、初期ノード数、初期アーク数は少量でよいとの判断によるものである。 M_3 を同じく 10 としたのは、初期ノード数が少量であるため、それ以上の数にすることができないためである。 M_4, M_5 の値は、ノードの追加よりもアークの追加を優先することで、ノードの有効性判定結果が効果的に働くであろうという判断から決定した。 $R = 7$ としたのは、HRP-1 の形状から予想される最大動作距離が 1 [m] 台であることから、最大 128 等分することで、最悪でも 1 [cm] 刻み程度の粒度が得られるとの判断によるものである。Fig. 8 にシミュレーション結果を示す。この例の動作計画には、約 2 分の平均計算時間を要した。ここに、単一の例に対して平均計算時間としたのは、ランダムなアルゴリズムであるため、利用した乱数の系列によって計算時間が異なるからである。この例の場合、100 回の試行を行って、その

Fig. 6 Layered architecture of the planning server

Fig. 5 Configuration of OpenHRP

Fig. 7 Initial configuration and final configuration

Fig. 8 Example of collision free motion

Table 2 Simulation results

case		nodes	valid	invalid	arcs	valid	invalid	time
0	ave	142.2	64.3	77.8	12868.3	5.0	493.0	123.7
	stdev	76.0	32.1	44.6	13185.4	0.9	412.2	118.4
1	ave	21.5	12.2	9.2	277.4	3.7	36.5	21.2
	stdev	10.9	5.1	6.6	306.7	0.7	35.9	7.0
2	ave	17.1	9.9	6.9	164.9	3.3	19.7	17.1
	stdev	7.6	3.4	5.0	206.7	0.5	20.1	5.1
3	ave	16.1	9.6	6.2	135.0	3.3	17.9	16.5
	stdev	5.3	2.2	4.1	104.3	0.5	11.6	3.3

平均計算時間を求めている。動作結果は、コンフィグレーション空間における離散的な点列で与えられるが、実際にヒューマノイドの運動に適用する場合にはこれを滑らかに補間して用いる。

四つのシミュレーション例についてそれぞれ100回の試行を行い、発生したノード総数、validと判定されたノードの数、invalidと判定されたノードの数、生成したアークの総数、validと判定されたアークの数、invalidと判定されたアークの数、総計算時間の平均値をTable 2に示す。また、それらの標準偏差も併記する。

それぞれのシミュレーションは、

- case0 障害物として穴のあいた壁面を用いる
- case1 case0の壁面の下部が欠けている
- case2 case0の壁面の下部および右端が欠けている
- case3 case0の壁面の上部のみが存在する

という四つのケースに対応する (Fig. 9)。

Table 2 から、生成したほとんどのノードに対して、有効性判定 $\lambda(n)$ が適用されていることが分かる。これは、最初に生成したノード数 M_1 が比較的小さく、かつ、この範囲で解が見つかったのがその理由である。これに対して、アークは生成されたアーク全体の一部についてのみ、有効性判定が適用されていることが分かる。例えば、case 0の場合、生成したアークの総数の平均は12868.3、有効性判定されたアークの平均数は $5.0 + 493.0 = 498.0$ であるので、判定処理が行われたのは5%

Fig. 9 Simulation setup (UpperLeft: case0 UpperRight: case1, LowerLeft: case2 LowerRight: case3)

未満である。このように、必要な部分にのみ有効性判定を適用しているので、計算時間の大きな削減が実現されている。

5. ま と め

本稿では、ヒューマノイドロボットの一般的な機構的特徴に着目して、全身バランス制御と共存可能な障害物回避動作計画アルゴリズムを提案した。提案した方法では、バランス制御による揺動が発生する場合でも、片腕の掃引体積を動作計画時と同一に保つことが可能である。

本手法は、動的なバランスを保持するために動作計画された軌道を変換する後処理を必要としない点に特徴があり、実時間フィードバック制御系との共存にも適しているものと考えられる。

今後は、バランス制御と共存した場合のダイナミクスシミュレーション、実験による検証を行っていく予定である。

参 考 文 献

- [1] J.J. Kuffner, S. Kagami, M. Inaba and H. Inoue: "Dynamically-stable motion planning for humanoid robots," In Proc. of 1st Int. Conf. on Humanoid Robotics (Humanoids'00), 2000.
- [2] H. Inoue, S. Tachi, K. Tanie, K. Yokoi, S. Hirai, H. Hirukawa, K. Hirai, S. Nakayama, K. Sawada, T. Nishiyama, O. Miki, T. Itoko, H. Inada and M. Sudo: "HRP: Humanoid Robotics Project of MITI," In Proc. of the First IEEE-RAS International Conference on Humanoid Robots, 2000.
- [3] Y. Nakamura, H. Hirukawa, K. Yamane, S. Kajita, K. Yokoi, K. Tanie, M.G. Fujie, A. Takanishi, K. Fujiwara, F. Kanehiro, T. Suehiro, N. Kita, Y. Kita, S. Hirai, F. Nagashima, Y. Murase, M. Inaba and H. Inoue: "V-HRP: Virtual Humanoid Robot Platform," In Proc. of the First IRRR-RAS International Conference on Humanoid Robots, 2000.
- [4] S. Kajita, K. Yokoi, M. Saigo and K. Tanie: "Balancing a Humanoid Robot Using Backdrive Concerned Torque Control and Direct Angular Momentum Feedback," Proc. of the IEEE International Conference on Robotics and Automation (ICRA), Seoul, Korea, May 2001, pp.3376-3382.
- [5] T. Yoshikawa: "Analysis and Control of Robot Manipulators

- with Redundancy," Int. Symp. Robotics Research, pp.735-747, 1987.
- [6] C.A. Klein and C.H. Huang: "Review of Pseudoinverse Control for the Use with Kinematically Redundant Manipulators," IEEE Trans. System, Man and Cyber, vol.13, no.3, pp.245-250, 1983.
- [7] Y. Nakamura, H. Hanafusa and T. Yoshikawa: "Task-Priority based Redundancy Control of Robot Manipulators," Int. J. Robotics Research, vol.6, no.2, pp.3-15, 1987.
- [8] I.D. Walker and S.I. Marcus: "Subtask Performance by Redundancy Resolution for Redundant Robot Manipulators," IEEE J. Robotics and Automation, vol.4, no.3, pp.350-354, 1988.
- [9] F-T. Chen, Y.D. Lu and Y.Y. Sun: "Window-Shaped Obstacle Avoidance for a Redundant Manipulator," Proc. IEEE Int. Conf. Robotics and Automation, pp.556-561, 1997.
- [10] L.E. Kavraki, P. Svestka, J-C. Latombe and M.H. Overmars: "Probabilistic roadmap for path planning in high-dimensional configuration spaces," IEEE Trans. Robotics and Automation, vol.12, no.4, pp.566-580, 1996.
- [11] J. Reif: "Complexity of the Mover's Problem and Generalizations," Proc. 20th IEEE Symp. FOCS, 1979.
- [12] J. Canny: The Complexity of Robot Motion Planning. The MIT Press, 1988.
- [13] Y. Nakamura, H. Hirukawa, K. Yamane, S. Kajita, K. Fujiwara, F. Kanehiro, F. Nagashima, Y. Murase and M. Inaba: "Humanoid robot simulator for the METI HRP Project," Robotics and Autonomous Systems, vol.37, pp.101-114, 2001.
- [14] [http://www.omg.org/Object Management Group](http://www.omg.org/Object%20Management%20Group).
- [15] S. Gottschalk, M.C. Lin and D. Manocha: "OBB-Tree: A Hierarchical Structure for Rapid Interference Detection," In Proc. Of ACM Siggraph '96, 1996.

藤原清司 (Kiyoshi Fujiwara)

1972年4月11日生。1997年筑波大学大学院工学研究科修了。同年通産省工業技術院電子技術総合研究所入所。2001年改組より独立行政法人産業技術総合研究所知能システム部門ヒューマノイド研究グループ研究員。生体制御，医療福祉工学，マンマシンインタフェース，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)

比留川博久 (Hirohisa Hirukawa)

1959年1月15日生。1987年神戸大学大学院自然科学研究科システム科学専攻修了。学術博士。同年通産省工業技術院電子技術総合研究所に入所。1994～1995年スタンフォード大学客員研究員。2001年改組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ヒューマノイド研究グループ長。ロボットの動作計画アルゴリズム，ネットワークロボティクス，ヒューマノイドロボット等の研究に従事。

(日本ロボット学会正会員)

金広文男 (Fumio Kanehiro)

1971年7月22日生。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年通産省工業技術院電子技術総合研究所入所。現在独立行政法人産業技術総合研究所知能システム部門ヒューマノイド研究グループ研究員。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，人工知能学会，日本ソフトウェア科学会，情報処理学会の会員。

(日本ロボット学会正会員)