離散的ボロノイ領域を用いた非凸多面体間の最短距離計算手法*

川地 克明**

鈴木 宏正***

Distance Computation between Non-convex Polyhedra Based on Discrete Voronoi Regions

Katusaki KAWACHI and Hiromasa SUZUKI

An algorithm for calculating the minimum distance between non-convex polyhedra is described. A polyhedron is represented by a set of triangles. In calculating the distance between two polyhedra, it is important to search efficiently the pair of the triangles which gives the pair of closest points. In this paper discrete Voronoi regions are prepared as voxels around non-convex polyhedra. Each voxel is given a list of triangles which have possibility to be closest to the voxel. When a triangle on the other object is intersecting a voxel, the closest triangles can be efficiently searched from this list. The algorithm has been implemented, and the results of distance computations show that it can calculate the minimum distance between non-convex polyhedra composed of a thousand of triangles at interactive rates.

Key Words: Computational Geometry, Distance Computation, Voronoi Region

1. はじめに

計算機を利用して機械機構の設計を行う場合、計算機上の3次元空間内に表現された部品が互いに干渉しているかどうかを検査する必要がある。また、このような部品間の干渉検査に加え、位置と姿勢が変化する部品間の最短距離を計算することができれば、計算機を用いた機構の運動シミュレーションなどにおいて、部品が衝突するまでの時間を予測する際に有用であると考えられる。

本研究では、三角形の集合によって表現された多面体(三角形面多面体)を最短距離計算の対象とする。4本以上の辺を持つ面については、これをあらかじめ幾つかの三角形に分割しておく。また、幾何的に隣接している三角形についても、隣接しているという位相的な情報はなくてもよく、例えば空間内に散乱した三角形群などに関しても距離計算が可能な手法を提案する。

1.1 関連研究

多面体間の最短距離計算手法は、closest feature testを用いた手法と、configuration spaceを用いた手法とに大きく分類することができる。

I-Collide ライブラリ¹⁾では、多面体を構成する要素である頂点・稜線・面(これらの要素のことを「フィーチャ」と総称する)のうち、最も近い距離にあるフィーチャの対を追跡し、そのフィーチャ対から物体上の最も接近した点の対を計算する。このアルゴリズムは、凸多面体上で最も接近したフィーチャ対に関する必要十分条件(closest feature test)を利用している。また、運動シミュレーションでの時間的に前後したフレームでは、最接近フィーチャの対は物体上を大きく移動しないことを利用して高速な最接近点探索を行っている。これを改良したV-Clipライブラリ²⁾では、最接近フィーチャの探索時にフィーチャ間の距離関数の微分値の符号を利用することで、より頑健かつ高速なアルゴリズムを実現した。V-Clipではまた、互いに干渉した凸多面体においても干渉部分の深さを最短距離として計算することが可能である。

また、凸多面体をその頂点群で代表される凸包として扱い、2

多面体の相対的な位置関係によって定義される configuration space 上を探索し、最短距離を与えるフィーチャ対を見つけ出す手法の一つとしてGJK アルゴリズム³⁾という手法が提案されている。

2. Voronoi 領域分割を利用した最短距離計算手法

Closest feature test を利用した最短距離計算手法では、多面体が凸であることを利用して多面体の周囲の空間を Voronoi 分割している。本研究ではこれを拡張し、非凸多面体の周囲の空間を、ボクセルを利用して離散的に Voronoi 分割して closest feature test を行うことによって最短距離計算を行う手法を提案する。

2.1 最短距離計算アルゴリズムの計算時間

三角形面多面体間で干渉検査を行う場合、その計算時間は一対の三角形間の干渉検査・距離計算に必要な時間と、干渉計算の対象となる三角形の組合わせの数とを掛け合わせたものである。

三角形間の干渉検査には複雑な計算が必要になるため、高速に干渉検査を行うためには、干渉検査をより簡単に行えるバウンディングボリュームを用いて三角形が占める空間を代表させ、三角形間の直接の干渉検査回数を減少させる手法が用いられる。このようなバウンディングボリュームを用いた代表的な干渉計算手法には、axis-aligned boxes, octrees⁴⁾, sphere trees⁵⁾, OBB-Tree⁶⁾, swept sphere volumes⁷⁾などが挙げられる。

これらの手法によって干渉検査を行う場合の計算時間 T は、以下の式(1)によって表すことができる⁸⁾。

$$T = N_v \times C_v + N_p \times C_p, \quad (1)$$

T : 干渉検査に必要な時間
 N_v : バウンディングボリューム間の干渉検査時間
 C_v : バウンディングボリューム間の干渉検査回数
 N_p : 三角形間の干渉検査時間
 C_p : 三角形間の干渉検査回数

多面体間の干渉検査に加えて最短距離の算出も行う場合には、もっとも単純なアルゴリズムでは2つの多面体に含まれる全ての三角形について、その全ての組み合わせに対して最短距離を

*原稿受付 平成13年2月10日

**学生会員 東京大学大学院工学系研究科(東京都文京区本郷7-3-1)

***正会員 東京大学大学院工学系研究科

計算し、その中で最も小さい距離を多面体間の最短距離とする。干渉検査を行う場合と同様に、三角形間の最短距離計算は複雑な処理であるため、距離計算の対象となる三角形の数を減少させることによってより高速な最短距離計算を行うことが可能である。

2.2 Closest Feature Test を用いた最短距離計算手法

凸多面体の最短距離計算手法である I-Collide¹⁾ および、これを改良した V-Clip²⁾ では

1. 多面体の周囲の空間を Voronoi 分割して closest feature test を行うことによって、最短距離を達成するフィーチャ対の探索を効率よく行う
2. 直前に得られた最接近フィーチャ対を探索の初期値として利用する (時間的連続性の利用)

というアルゴリズムを使用している。

これに対して、非凸多面体間の最短距離を計算する場合には、必ずしも最接近点の位置の時間的な連続性を利用することができないわけではない。したがって、非凸多面体における最短距離の計算速度は最接近フィーチャ対の探索をいかに高速に行うかにかかっていると言える。

2.3 離散的 Voronoi 分割

多面体の各面に対する Voronoi 領域の定義は以下の通りである：

定義 ある面 F に対する Voronoi 領域とは、多面体上の他のどの面への距離よりも F への距離の方が小さい点の集合である

例として、2次元平面上で凸物体の周囲に定義される Voronoi 領域の様子を Fig. 1 に示す。 R_a, R_b, R_c はそれぞれ辺 a, b, c に対応する Voronoi 領域である。このとき、別の物体上の辺 x が領域 R_b に含まれていれば (Fig. 2)、物体上で辺 x に最も近い辺は b であり、 x と a, c との最短距離を実際に計算する必要がないことがわかる。

凸多面体に対してはこのような Voronoi 領域を簡単に計算することができ、I-Collide¹⁾ や V-Clip²⁾ では Voronoi 領域を用いて凸多面体間の最短距離計算を行っている。

ここで、もし非凸多面体に対してもその周囲の空間を Voronoi 領域に分割することができれば、別の多面体上の面 x は、 x が含まれる Voronoi 領域に対応している多面体上の面に最も近いことがわかる。しかし、非凸多面体に関する Voronoi 領域分割の構造を正確に計算することは困難である。そこで、本研究では Voronoi 領域の構造を陽には計算せず、多面体の周囲の空間をボクセルに分割することで離散的な Voronoi 領域を定義する。分割されたボクセルの一つ一つには、ボクセルに最も近い可能性のある面のリストを与える (Fig. 3)。このリストを Closest Feature List (CFL) と呼ぶ。もし別の多面体上の面 x がボクセル V と共通部分を持てば、 x は V に与えられた CFL の面のいずれかと最も近いと言える。

2.4 最短距離計算の対象となる多面体

本研究では、以下に述べる条件を満たす多面体の間の最短距離を計算する手法を提案する：

- 多面体は三角形の集合として表される。Closest feature algorithm では多面体は3つのフィーチャ (面・稜線・頂点)

Fig. 1: 凸多角形の周囲の Voronoi 領域分割

Fig. 2: Voronoi 領域と最も近い辺

ab	ab	b	b	b	bc	bc
a	ab	b	b	bc	bc	c
a	a	ab	b	bc	c	c
a	a	ab	abc	bc	c	c

Fig. 3: ボクセルに与えられた CFL (Closest Feature List)

の集合として表されたが、本研究では面と稜線は独立したフィーチャとせず、三角形を全ての単位として扱う。

- 隣接している三角形間には必ずしも位相的な構造の情報を必要としないが、もし位相構造がある場合には、三角形を CFL を含む可能性のないボクセルを排除するために利用することができる。
- 多面体内の三角形は互いに干渉していてもよい

このような条件により、境界のある (穴が空いている) 多面体や、位相構造を持たない乱雑な三角形群などの形状を距離計算の対象として扱うことが可能となる。また、三角形面の表と裏を区別しないので、本手法によって計算される最短距離は常に 0 または正の値となる。

2.5 ボクセルに最も近い三角形

ボクセルに対して CFL (Closest Feature List) を与える際には、ボクセル内の点と三角形との最短距離の最大値・最小値を用いる。点 x と三角形面多面体が与えられた時、多面体内で x に最も近い三角形は x と多面体との相対的な位置関係によって変化するが、あるボクセル V に対する CFL は、 V 内の全ての点において、点に対して最も近い三角形を列挙することによって定義できる。以下にその手続きを示す。

ある三角形 A と点 x との最短距離を $d_A(x)$ と表す。ボクセル V の領域内には、 $d_A(x)$ を最大および最小にする点 $x_A^{\min}$ と $x_A^{\max}$ が存在する (Fig. 4)。この定義により、ボクセル V 内の全ての点に対して、以下の式 (2) が成立する：

$$d_A(x_A^{\min}) \leq d_A(x) \leq d_A(x_A^{\max}), \quad \forall x \in V \quad (2)$$

同様に、もう一つの三角形 B に対しても点との最短距離 $d_B(x)$ を定義でき、以下の式 (3) を満足する $x_B^{\min}$ と $x_B^{\max}$ とが存在する：

$$d_B(x_B^{\min}) \leq d_B(x) \leq d_B(x_B^{\max}), \forall x \in V \quad (3)$$

ここで、ある $x_A^{\max}$ と $x_B^{\min}$ とが $d_A(x_A^{\max}) < d_B(x_B^{\min})$ なる関係を満たせば、式 (2, 3) の定義により、以下の関係式 (4) が導かれる。

$$d_A(x) < d_B(y), \forall x \in V, \forall y \in V \quad (4)$$

すなわち、 V 内のどの点についても、三角形 B よりも三角形 A の方がより小さい最短距離を与え、従って B はボクセル V 内のどの点に対しても最も近い三角形とは成り得ない。

Fig. 4: 三角形とボクセル内の点との最短距離

ボクセル V に関する CFL を列挙する際には、まずはじめに多面体に含まれる全ての三角形に対して、三角形 i とボクセル V 内の点との距離 d_i の最大値 $d_i^{\max}$ と最小値 $d_i^{\min}$ を計算する。次に、三角形を距離の最小値によって並び替える。もしいくつかの三角形の最小値が等しい場合には、これをさらに距離の最大値によって並び替える。以上の手続きによって、最も小さい距離 $d_k^{\min}$ と $d_k^{\max}$ を与える三角形 k を決定できる。式 (4) より、条件 $d_k^{\max} < d_i^{\min}$ を満たす三角形 i は V 内の点に対して最も近くなることはないで、 V の CFL には含まれない。

本研究では、ボクセルは三角形面多面体の多面体のローカル座標系に対して定義し、最短距離計算のための CFL の作成は多面体に対して一度だけ行う。CFL は多面体の幾何情報と共に保存・再利用し、物体間の相対的な位置関係が変わった場合にも再び CFL を作成する必要はない。

本研究での実装では、多面体のバウンディングボックスの 2 倍の大きさの領域に対してボクセル分割を行っている。バウンディングボックスは x, y, z 軸に垂直な面で囲まれる直方体である。もし物体がこの領域を外れた場合、最短距離は物体の凸包間での最短距離によって近似する。

3. CFL による最短距離計算

ボクセル上の CFL を用いた三角形面多面体 A, B 間の最短距離の計算は、4 段階の手続きに分けることができる。

始めに、多面体 A 上のボクセルのうち、多面体 B の三角形と重なりあうものを列挙する。この手続きは、ボクセル群に対しては BSP 木を、多面体上の三角形に対しては OBBTree⁶⁾ をあらかじめ定義しておくことで効率よく行うことができる。

次に、重なり合うボクセルと三角形の対に対して closest feature test を行う。もし三角形 X と Y とが凸多面体上にあった場合には、

$$\begin{cases} \text{三角形 } X \text{ が三角形 } Y \text{ の Voronoi 領域に含まれる} \\ \text{かつ} \\ \text{三角形 } Y \text{ が三角形 } X \text{ の Voronoi 領域に含まれる} \end{cases} \quad (5)$$

という条件を満たしていることが、 X, Y 上に最接近点が存在するための必要十分条件となる。

非凸多面体に対して CFL を用いて最短距離計算を行う場合には、これを以下の条件 (6) として表すことができる：

$$\begin{cases} \text{三角形 } X \text{ とボクセル } V_Y \text{ とが重なり合う} \\ (V_Y \text{ の CFL は三角形 } Y \text{ を含む}) \\ \text{かつ} \\ \text{三角形 } Y \text{ とボクセル } V_X \text{ とが重なり合う} \\ (V_X \text{ の CFL は三角形 } X \text{ を含む}) \end{cases} \quad (6)$$

式 (6) は、非凸多面体に関しては十分条件ではないが必要条件となるため、この条件 (6) を満たさない三角形は、多面体間の最も近い三角形となる可能性はない。

次に、上の条件を満たすボクセル群を最短距離の最小値 $d_k^{\min}$ によって並び替え、最後に、ボクセルの CFL 内の三角形と、ボクセルと重なり合う三角形との最短距離を実際に計算する。

ボクセルを最短距離によってソートすることにより、最後の段階での三角形間の最短距離計算において、余分なボクセルを計算の対象から除外することができる。計算を進めるうち、すでに得られた三角形間の最短距離よりも大きな $d_k^{\min}$ を持つボクセルに行き当たった場合、これ以降のボクセルは最短距離を与える可能性がないため、その時点で処理を打ち切ることができる。

3.1 隣接関係を利用した CFL の削減

多面体上での三角形間の接続関係が利用できる場合には、三角形上に近似的な Voronoi 領域を定義することで、ボクセル上の CFL をより少なくすることができる。この近似的 Voronoi 領域は、三角形の辺と、三角形の辺の両側の三角形の平均法線とを含む平面によって囲まれる領域とする。この領域を 2 次元平面での概念図によって示したものが Fig. 5 中の破線で囲まれた灰色の領域である。

凸多面体の場合には、この近似による Voronoi 領域は正確な Voronoi 領域と一致する。非凸多面体の場合には、近似による領域は正確な Voronoi 領域を含むので、CFL が不完全になることはない。

Fig. 5: 近似的ボロノイ領域によるボクセルの削減

4. 結果

前節までで述べた離散的ボロノイ領域と CFL による手法を使用して、非凸三角形面多面体間の最短距離を計算するプログラムを実装した。

4.1 CFL の長さ

1314 面の三角形面多面体 (Fig. 6) に関して、ボクセルの分割数と 1 ボクセル当りに与えられる CFL の長さの平均値との関

係を表1に示す。CFLの平均長はボクセルが小さいほど短くなり、より小さなボクセルを用いることで効率的な最短距離計算が可能になると言える。このとき、ボクセルの大きさが三角形の大きさよりも小さくなるとCFLの平均長はあまり小さくならないことがわかっていく。

実際の最短距離計算に使用される三角形の変化をFig. 7に示す。図中の暗い色で塗り潰された三角形がCFLによって最短距離計算の対象となった三角形であり、物体間の太い線分の両端が多面体間の最短距離を与える点を示している。

ボクセル上のCFLの算出は最短距離計算の前処理として行われる。計算されたCFLの例をFig. 8に示す。多面体の背中の部分の黒く塗り潰された三角形は、黒い辺で縁取られたボクセルのCFLに含まれている。

Table 1: ボクセル分割数とCFLの平均長

ボクセル分割数	CFLの平均長
$10 \times 10 \times 10$	29.08
$15 \times 15 \times 15$	11.92
$20 \times 20 \times 20$	10.90
$30 \times 30 \times 30$	6.35

4.2 CFLを用いて計算された最短距離

$30 \times 30 \times 30$ に分割したボクセルを用いてCFLを定義し、1314面の三角形面多面体間で最短距離計算を行った場合には、実際に最短距離の計算を行った三角形の対の数は平均して約172対だった。これは、全ての三角形の対の組み合わせの数に比べて、約10000分の1の数である。1回の最短距離計算に必要な平均CPU時間は、MIPS R10000 (250Hz)を用いた場合で0.061秒となった。計算された最短距離(図中太線)と最短距離に使用された三角形(黒く塗られた三角形)が、物体の移動によって変化する様子をFig. 9に示す。

4.3 他手法との比較

Table 2に、本研究での手法とPQPライブラリ⁷⁾とによる最短距離計算に必要な時間の比較を示す。比較にはFig. 6に示す多面体を用い、CFLは $30 \times 30 \times 30$ に分割したボクセルに対して定義した。PQPライブラリはswept sphere volumes⁷⁾を用いて干渉検査を行い、距離計算を干渉検査に帰着させるアルゴリズムによって、非凸多面体間の最短距離計算を行うことも可能にしている。本研究の実装はPQPの約3.4倍の計算時間を要する。これは主に N_v の回数が大きことに起因している。すなわち、3節で述べた計算手順で三角形とボクセルとの干渉検査の回数が多いことを示しており、実際には計算時間の70%以上をこの干渉検査に費やしている。より効率的な最短距離計算を行うためにはボクセルとボクセル内のCFLについて階層的な構造を生成することによって干渉検査の回数 N_v を減少させる手法が考えられる。

Table 2: 最短距離計算に要するCPU時間の比較

手法	平均計算時間	N_p の平均値	N_v の平均値
PQP	0.018 秒	105.4	1781
CFL	0.061 秒	171.6	7970

Fig. 6: 非凸三角形面多面体(左から1314面, 1140面, 500面)

Fig. 7: ボクセル分割数と計算に使用される三角形の変化

5. 結 論

非凸多面体間の周囲の空間を離散的ボロノイ分割し、CFL (Closest Feature List) を利用して多面体間の最短距離を計算する手法を示した。また、ボクセルと三角形群との距離の最大値・最小値を利用することにより、最短距離計算の必要のないボクセルを効率よく排除できることを示した。ボクセル当りのCFLの平均長はボクセルの大きさによって変化するもので、より効率的な距離計算のためには、三角形面多面体の幾何的な特徴に応じてボクセルのサイズを調節する手法が必要であると考えられる。

本研究での手法では、多面体同士が互いに干渉している場合には干渉を検知して、最短距離として0を返す。アプリケーションとして運動シミュレーションを考えた場合には、衝突が発生する時間を推定するために、干渉が発生している場合には負の距離を計算できることがより望ましい。

CFLの作成は時間のかかる処理であり、例えば1314面の三角形面多面体に対して $30 \times 30 \times 30$ に分割したボクセルを定義し、MIPS R10000 (250MHz)を用いた場合の計算時間は約40分を要する。しかし、近年の高速なグラフィクスハードウェアによるVoronoi領域の計算手法⁸⁾を利用すれば、CFLの作成はより高速に行うことが可能であると考えられる。

Fig. 8: 三角形を CFL に含むボクセル群

参考文献

- 1) Ming C. Lin and John F. Canny. Efficient algorithm for incremental distance computation. In *IEEE Conference on Robotics and Automation*, 1991.
- 2) Brian Mirtich. V-Clip: Fast and robust polyhedral collision detection. Technical Report TR-97-05, MERL, July 1997.
- 3) Elmer G. Gilbert, Daniel W. Johnson, and S. Sathiya Keerthi. A fast procedure for computing the distance between complex objects in three-dimensional space. *IEEE Journal of Robotics and Automation*, Vol. 4(2), pp. 193–203, April 1988.
- 4) H. Samet. *Spatial Data Structures: Quadtree, Octree and Other Hierarchical Methods*. Addison Wesley, 1989.
- 5) Joe Pitt-Francis and Roy Featherstone. Automatic generation of sphere hierarchies from cad data. In *Proceedings of IEEE International Conference on Robotics and Automation*, 1998.
- 6) S. Gottschalk, M. C. Lin, and D. Manocha. OBBTree: A hierarchical structure for rapid interference detection. In *Computer Graphics Proceedings, Annual Conference Series*, pp. 171–180. ACM SIGGRAPH, 1996.
- 7) Eric Larsen, Stefan Gottschalk, Ming Lin, and Dinesh Manocha. Fast proximity queries with swept sphere volumes. Technical Report TR99-018, Department of Computer Science, University of North Carolina, Chapel Hill, 1999.
- 8) Kenneth E. Hoff III, Tim Culver, John Keyser, Ming Lin, and Dinesh Manocha. Fast computation of generalized voronoi diagrams using graphics hardware. In *Computer Graphics Proceedings, Annual Conference Series*, pp. 277–285. ACM SIGGRAPH, 1999.

(1–4), (a–e): 最短距離計算の連続写真

黒い三角形: 距離計算に使用された三角形
太線: 最短距離にある点の対

Fig. 9: 移動する非凸多面体間の最短距離