

超3角形BRepにおけるEdge-basedデータ構造と形状演算アルゴリズム

荒川 佳樹[†] 山口 富士夫^{††}

Boundary Representation (BRep) において、境界面を3角形面のみで構成すると、データ構造および処理アルゴリズムともに究極的に単純となるが、データ量(3角形の数)の大幅な増大を招く。このような課題を解決するために、3角形の3つの頂点が同一直線上となるゼロ3角形を導入した超3角形BRepを以前に提案した^{(14),(15)}。このように3角形を拡張することにより、3角形処理の致命的欠点である3角形の数の増大を抑制することに成功した。従来の超3角形BRepではFace-basedデータ構造を採用していた。本論文では、超3角形処理に適した、究極的に単純なEdge-basedデータ構造であるトライアングルエッジを提案する。そして、このデータ構造を用いた新しい形状演算アルゴリズムもあわせて提案する。すなわち、従来別処理であった3角形分割処理と境界面接続処理を融合した、より簡略化されたアルゴリズムを提案する。また、計算機実験により、本提案アルゴリズムは、交差部分が多い形状に対して、特に有効であることを実証した。

An Edge-based Data Structure for Extended Triangle-based BRep and Its Set Operation Algorithm

YOSHIKI ARAKAWA[†] and FUJIO YAMAGUCHI^{††}

Triangle-based BRep is the simplest among the BRep models in terms of algorithms and data structures. It has a fatal weak point, however: the number of triangular faces increases dramatically when set operations are carried out. In the past we abated this problem by introducing a so-called Zero triangle, whose three vertices are collinear. This Zero triangle-based BRep used a face-based data structure. In this paper we propose a simple edge-based data structure (so-called Triangle Edge) suitable for the Zero triangle-based BRep. We also propose a new algorithm for set operations that uses the proposed Triangle Edge. We achieve a simplified algorithm by unifying the splitting process of triangles and the connecting process of boundary faces, which were separate in the previous algorithms. Computational experiments show that this proposed algorithm performs much faster than the previous ones in the case where there are many intersectional parts between shapes.

1. はじめに

Boundary Representation (BRep) では、その境界面は多角形面(以下、単に多角形と呼ぶ)を用いて通常表現/処理される^{(1),(3),(8),(9)}。しかし、多角形に基づいたBRepはデータ構造および処理アルゴリズムともに複雑となり、その処理系も大規模なものとなる。一方、3角形は究極的に単純な根元的基本図形である。そこで、3角形面(以下、単に3角形と呼ぶ)

を用いてBRepを構成すると、非常に単純な形状表現となり、データ構造および処理アルゴリズムは、ある意味で究極的に単純化・簡略化される。しかし、処理の過程でデータ量(3角形の数)の大幅な増大を招くなど大きな欠点を持つ。

そこで、多角形処理と3角形処理を併存させて、両者の優れた特性を生かしたハイブリッドモデルが提案されてきている^{(4),(11)~(13)}。しかし、このようなハイブリッドモデルでは、多角形を3角形に分割する処理および3角形から多角形を合成する処理が絶えず発生する。また、データ構造的に見ても、多角形データと3角形データが併存することとなる。

「3角形処理の究極的単純性」は非常に魅力的であり捨て難い。そこで、筆者らは3角形を拡張した超3角形を考案することにより、上記のような3角形処理

[†] 郵政省通信総合研究所通信システム部非常時通信研究室
Multimedia Communications Section, Communication
Systems Division, Communications Research Laboratory,
Ministry of Posts and Telecommunications

^{††} 早稲田大学理工学部機械工学科
Mechanical Engineering Department, Waseda University

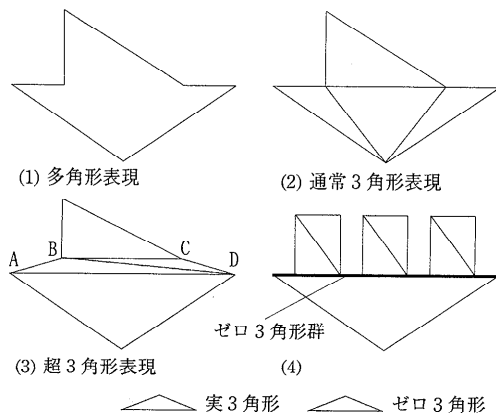


図1 ゼロ3三角形と実3三角形
Fig. 1 Zero triangle versus real triangle.

が持つ単純性をほとんど損なわずにその欠点を解消した。そして、超3三角形を用いた**超3三角形BRep**を提案した^{(14),(15)}。超3三角形は、図1(3)に示すように、3つの頂点が同一直線上にあり面積がゼロとなる3三角形である**ゼロ3三角形**と、3頂点が同一直線上とならない通常の3三角形である**実3三角形**から構成される。図1(3)では頂点A, B, C, Dは同一直線上にある。図1(3), (4)に示すように、ゼロ3三角形を用いることにより、ゼロ3三角形の数は増えるが、より少ない実3三角形(通常3三角形)で面分を表現することができる。このような特徴により、超3三角形処理では、3三角形の表現性とその処理性を飛躍的に増大させることができる。

超3三角形BRepに関しては、これまでに、その基本的な概念と概要⁽¹⁴⁾、および交線データを用いた高速形状演算アルゴリズム⁽¹⁵⁾を提案した。これらはいずれもデータ構造としてはFace-basedデータ構造を採用している。

本論文ではBRepにおけるデータ構造を論じ、3三角形処理の持つ究極的単純性を損なわない単純なEdge-basedデータ構造であるトライアングルエッジを提案する(2章)。また、このトライアングルエッジを用いた、より簡略化されたかつ効率的な新しい形状演算アルゴリズムもあわせて提案する(3章)。そして、この新しいアルゴリズムと、従来の交線データを用いた高速形状演算アルゴリズムとを計算機実験により比較評価する(4章)。

2. Edge-based データ構造

BRepにおけるデータ構造は、

- (1) Edge-based データ構造
- (2) Face-based データ構造

に大別することができる⁽⁵⁾。前者はエッジ(辺または

稜線)を主体とした、後者は面を主体としたデータ構造である。そして、BRepではこれまで圧倒的に前者のEdge-basedデータ構造(以下、Eデータ構造と呼ぶ)が採用されてきている。その主な理由は、コンパクトなデータ構造になり、かつデータ処理効率が良いからである。しかし、幾何演算誤差などの管理は容易ではない。これは、面、エッジ、頂点データを対等に扱い、面データを第一義データとして扱うことが難しいからである。

これに対して、Face-basedデータ構造(以下、Fデータ構造と呼ぶ)は以下のような特徴を持っている⁽⁵⁾。

- (1) 設計・エンジニアリングデータベースなどのより上位のデータとの親和性が良い。たとえば、形状特徴情報を面の拘束として直接的に記述できる。
- (2) 面データを第一義データとすることができ、幾何演算誤差などの管理がやりやすい。

ここで、BRepにおいてこれまでに提案されている主なEデータ構造をまとめておく^{(2),(10),(16),(17)}。

1. Winged edge (W-E)

このデータ構造は図2(1)に示すように、エッジを中心とした、それに隣接する頂点、エッジ、および面分の接続関係を記述している。すべてのデータを1つのデータ内に保持しているので、データ数が少なくてすむ。しかし、次のような欠点を持つ。

- (1) 面分/頂点まわりの探索処理では判定処理(if文)が必要となりアクセス効率が悪い。
- (2) 多重連結の面分には対応できない。
- (3) 双対性が欠如している。

2. Half edge (H-E)

このデータ構造は、図2(2-1)に示すように、エッジが2つに分けられ、1つのエッジデータは片方の端点(頂点)と片方の隣接面の情報しか持たない。なお、図2におけるカッコの付いているポイントは理論上必要ないものであるが、アクセスを高速にするために通常追加される。H-Eには、図2(2-1), (2-2), (2-3)に示すように、面分まわり(FE型)、頂点まわり(VE型)、およびこの両者を併合した形の3種類が考えられる。H-Eは1つのエッジを2つに分けて表現するので、W-Eに比べて保持するデータ量は多くなるが、判定処理が不要となるため、アクセス処理は簡略化・効率化される。

3. Quarter edge (Q-E)

さらに、エッジを4つに分ける方法が考えられる。図2(3)の一方は面分まわり、他方は頂点まわりとして使われる。この対になっている2個のQ-Eを合わせ

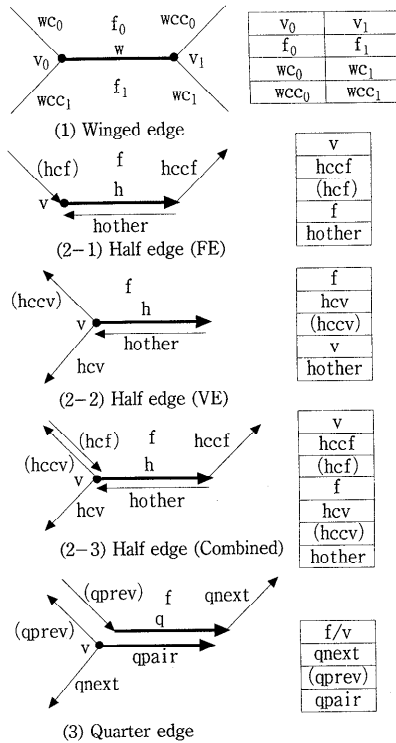


図2 Edge-based データ構造
Fig. 2 Edge-based data structures.

ると1個のH-Eと同じ役割を持つ。このように、Q-Eは面まわりと頂点まわりの処理を完全に分離することができるので、双対性を持ったデータ構造となる。しかし、Q-Eでは、H-Eに比べて、ポイントの数が増え全体のデータ量が多くなる。また、ポイントの参照回数が多くなるため、H-Eに比べてアクセス効率が悪くなる。

筆者らが提案した超3 角形 BRep では、これまで F データ構造を採用してきた¹⁴⁾。これまでの超3 角形 BRep のデータ構造は図3(1)に示すように、3 角形面データと頂点データから構成される。このように、超3 角形 BRep の F データ構造は非常に単純となる。エッジデータは、特に陽に持つ必要はない。

次に、超3 角形処理における E データ構造に関して考察する。まず、超3 角形処理において、面分としては3 角形面しか存在しない。したがって、3つのエッジで囲まれる3 角形面とその隣接関係のみが表現できれば必要十分である。そこで、W-EやH-Eの併合型のように多くのデータ項目を持つ必要はない。また、3 角形処理では、処理アルゴリズムは究極的に単純化・簡略化されるので、双対性を利用する効果はかなり低下する。さらに、3 角形処理では多角形処理に比べて

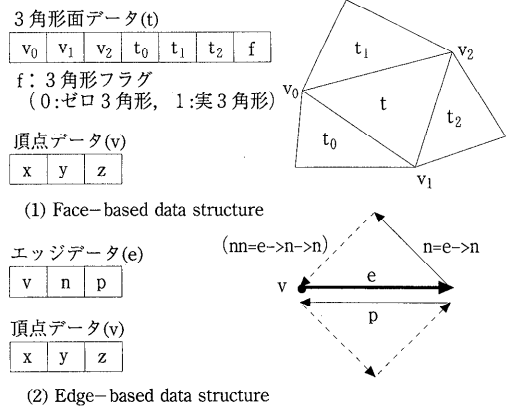


図3 超3 角形 BRep におけるデータ構造
Fig. 3 Data structures on extended triangle-based BRep.

一般的にデータ量が増えるので、データ項目の数はできるだけ少ない方がよい。したがって、Q-Eのようなポイントを多用する、データ量が増えるデータ構造は適さない。

これらの条件を満たすものとして、図3(2)に示すデータ構造を提案する。これはH-EのFE型をベースに冗長な項目を削った、必要最小限の構成である。任意の多角形を表現する場合は、後ろから接続するエッジ情報nnはデータアクセスの効率化に貢献するが、3 角形の場合はこのエッジはe→n→nというように簡単にアクセスできるので不要である。このデータ構造をトライアングルエッジと呼ぶことにする。

3. 新しい形状演算アルゴリズム

3.1 概要

これまでに発表した2つの形状演算のアルゴリズムの大枠は、いずれも以下の5つの処理ステップから構成される^{14),15)}。

- (1) 共通空間処理. 形状演算を行う2つの形状がともに存在する共通空間 (Bounding box) を求める。
- (2) 分割処理. この共通空間内において、両形状の実3 角形が交差するかどうかのチェックを順次行い (図4(1))、交差する場合は交線を求めて、その交線において3 角形を分割する (図4(2))。
- (3) 消去処理. 不要となる3 角形を消去する (図4(3))。
- (4) 接続処理. 交線において接する両形状の3 角形を、ゼロ3 角形を用いて接続する (図4(4))。
- (5) 統合処理. 隣接する周辺の3 角形との形状チェックを行い、それらがより大きな1つの3 角形を

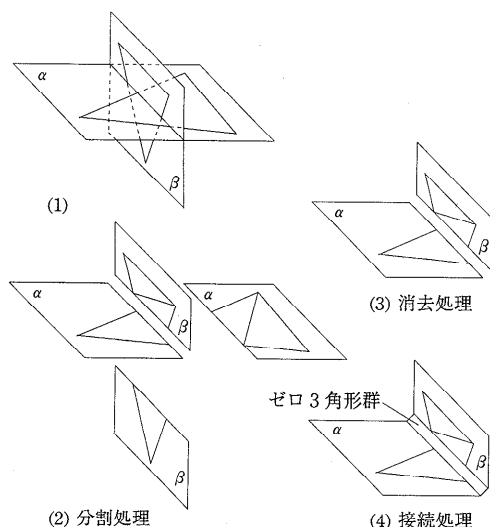


図4 従来の形状演算アルゴリズム

Fig. 4 Previous Boolean shape operation algorithm.

形成する場合は3三角形の統合を行う。

文献 14) と文献 15) におけるアルゴリズムの違いは、後者が、交線データという付加的データを導入することにより、処理が効率化・高速化されている点にある。交線データは分割処理が行われたときに、互いに接する両形状の3三角形に関する情報を保持する。そこで、接続処理において、この情報を利用することにより処理が効率化される。

ここでは、2章で提案した、Eデータ構造の1つであるトライアングルエッジを用いた、より簡略で効率的な形状演算アルゴリズムを提案する。トライアングルエッジでは、隣接する3三角形のエッジへのポイントを保持しており、このポイントを変更するだけで、隣接3三角形の変更すなわち境界面接続が簡単かつ効率良く行える。そこで、この優れた特徴を利用して、図5に示すように、分割処理(図5(2-1))を行った直後に、接続処理(図5(2-2))を行う。この統合化された処理を「分割接続処理」と呼ぶことにする。

この新しい処理では、ゼロ3三角形が重要な役割を演じる。すなわち、実3三角形の分割は、ゼロ3三角形を用いて分割が周辺の3三角形に及ばないように局所的に行う(図5(2-1))。また、これらの分割された両形状の3三角形の接続も、ゼロ3三角形を用いて局所的に行う(図5(2-2))。この方法により、従来のように、接続する3三角形に関する情報を保持する必要がなくなる。すなわち、従来のアルゴリズムにおいて用いていた交線データおよびそれに関連する処理がまったく不要となり、処理の効率を犠牲にすることなく処理が簡略化さ

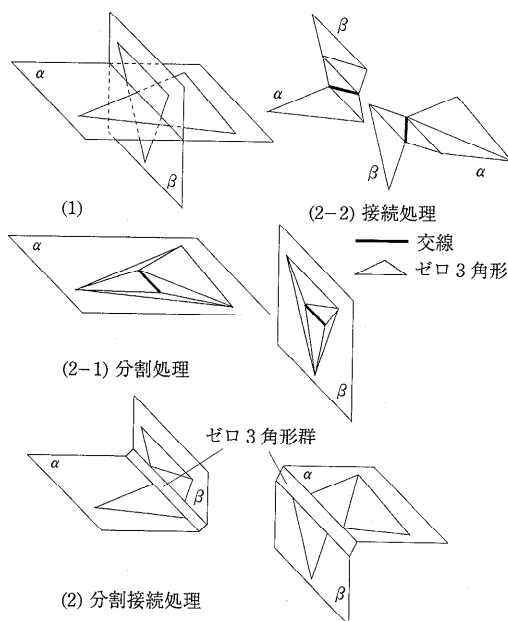


図5 新しい形状演算アルゴリズム

Fig. 5 Proposed Boolean shape operation algorithm.

れる。

新しいアルゴリズムでは、従来のアルゴリズムと比べて、この分割接続処理だけが根本的に異なる処理となる。そこで、以下この処理に関して説明する。ここでは、形状 α と形状 β との形状演算を想定する。新しい分割接続処理ではまず、両形状の実3三角形に対して交差判定処理(3.2.1項参照)を行う。そして、交差する場合は(図5(1)参照)、その交線の生じる位置に応じて、3つのパターンに分類される(図6参照)。次に、このそれぞれの交差パターンに応じた処理(3.2.2~3.2.4項参照)を行う。以上の処理を両形状の実3三角形のすべての組合せに対して行う。この結果として、図5(2)に示すように、境界面の分割と接続が完了する。そして、この分割接続処理の後に、不要境界面の消去処理が行われる。

新しいアルゴリズムでは3三角形の分割後すぐに(局所的に)接続を行うが、この接続処理を簡潔かつ矛盾なく行うために、ここでは、「3三角形面(境界面)の接続に用いるエッジ上(両端点は除く)には頂点は存在しない」という接続エッジルールを設けた。このルールを用いた処理概要は次のようになる。まず、両3三角形の「共通交線」を求める(図6参照)。次に、この共通交線上に頂点がある場合は、その頂点において交線を分割する。すなわち、交線上に頂点が存在しない「部分共通交線」を求める(図8, 9参照)。そして、この部分共通交線と一致するエッジを生成することに

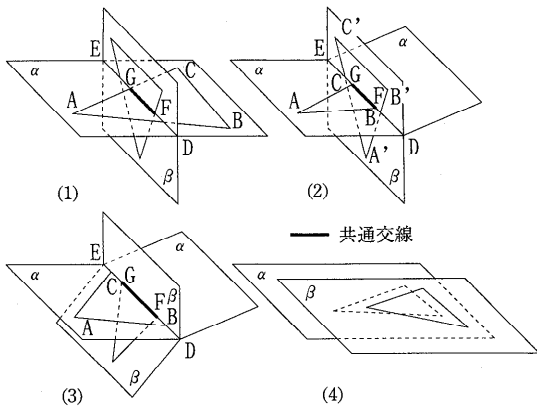


図6 実3三角形の交差パターン

Fig. 6 Intersectional patterns of two real triangles.

より、上記の接続エッジルールに合致した、すなわち、両形状において両端点を共有する（まったく一致する）エッジのペアを生成することができる。これにより、エッジの接続処理が非常に単純化される。以下、これらの処理の詳細に関して説明する。

3.2 分割接続処理

3.2.1 交差判定処理

形状 α の実3三角形と形状 β の実3三角形を順次取り出し、両者の交線を求めることにより、交差判定を行う。交差する（交線が存在する）場合は、さらに、この交線と3三角形面の位置関係により、図6(1), (2), (3)に示す3通りに分類される。図6(4)に示す両3三角形が重なる（同一平面となる）場合など、これら以外の場合は3三角形の分割および接続処理は行われない。以下、この3つのケースにおける処理内容についてそれぞれ説明する。なお、形状 α の3三角形における処理を中心に説明するが、形状 β の3三角形に関しても同様の処理となる。

3.2.2 交差する場合

図6(1)に示すように、形状 α に属する実3三角形ABCと、形状 β に属する実3三角形とが交差する場合である。この場合にはまず、交線DEを求め、この交線において両者の3三角形のどちらにも含まれる部分FGを求める。これを「共通交線」と呼ぶことにする。次に、この共通交線と一致するエッジを生成するために、両形状の3三角形をさらに「位相分割」（後述）する。この位相分割処理は、図7(1), (2), (3)に示す3通りに分類される。図7では形状 α の3三角形ABCの場合を示しているが、形状 β の3三角形の場合もまったく同様となる。

図7(1)の場合は共通交線FGとエッジ（3三角形AFG

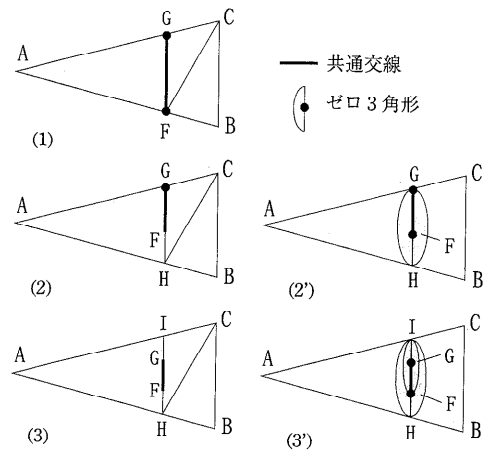


図7 ゼロ3三角形による分割パターン

Fig. 7 Split patterns of a triangle by using zero triangles.

の辺FG)がまったく一致する場合である。この場合は位相分割処理は不要である。(2)の場合は共通交線の片端点のみがエッジの片端点と一致する場合である。この場合は、図7(2')に示すように、2つのゼロ3三角形FGHとFHGを用いて位相分割を行い、頂点Fおよび共通交線FGと一致するエッジFGを生成する。位相分割とは、このようにゼロ3三角形を用いて頂点とエッジを増やすために行う位相的な分割であり、通常の3三角形分割とは異なる。(3)の場合は共通交線の両端点ともにエッジの端点と一致しない場合である。この場合は、図7(3')に示すように、4つのゼロ3三角形FIH, FHI, GIFおよびGFIを用いて位相分割を行い、頂点F, GおよびエッジFGを生成する。

以上のような位相分割を両形状の3三角形に関して行うと、両形状において共通交線と一致するエッジがそれぞれ生成される。これにより、図5(2-2)の接続処理に示すように、共通交線と一致する両形状のエッジを接続することにより、両形状の面を局所的に接続することができる。このように、2つの実3三角形が交差する場合の分割接続処理は、非常に単純となる。

3.2.3 3三角形面上となる場合

図6(2)の例のように、形状 β の実3三角形面上に、形状 α の実3三角形の1つのエッジが乗る場合である。もちろん、これとは逆に、形状 α の実3三角形面上に形状 β の実3三角形が乗る場合もこれに含まれる。ここでは前者の場合の処理に関して説明する。この場合は、以下のステップ1~4の処理を行う。

(ステップ1) 共通交線分割処理

図6(2)の形状 α の3三角形ABCの周辺を図示すると、たとえば図8(1)となる。図8(1)では、点B, C,

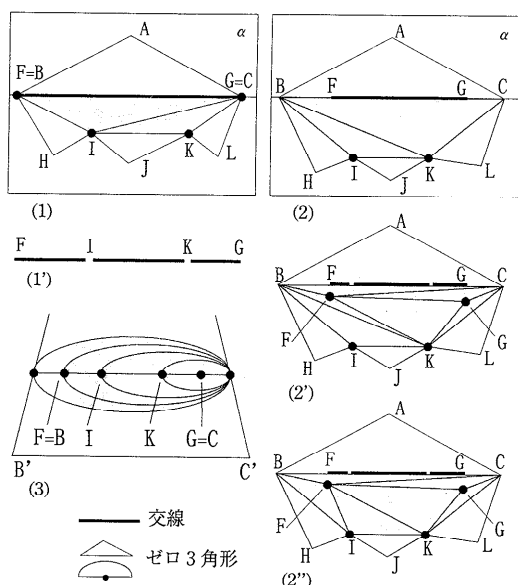


図8 共通交線処理

Fig. 8 Processing for a common intersectional line.

F, G, I, K はすべて同一直線上にある。そして、3つの実3角形 BHI, IJK, KLC が実3角形 ABC に辺 BC を介して隣接している。この例のように共通交線上に頂点が存在する場合は、共通交線を頂点において分割する処理を行う。まず、 α 面において共通交線 FG 上にある頂点をすべて求める。図 8(1) の例では、頂点 B, I, K, C が共通交線 FG 上の点として求められる。ここで、F と B、および G と C は同一点である。次に、これらの頂点において、共通交線 FG を「部分共通交線」に分割する。図 8(1) の例では、共通交線 FG は、部分共通交線 FI, IK, KG に分割される (図 8(1') 参照)。また図 8(2) の例の場合は (ここでも、頂点 I, K は交線 FG 上にある)、共通交線 FG 上の頂点として I と K が求められ、部分共通交線 FI, IK, KG に分割される (図 8(2') 参照)。

(ステップ 2) 面交差判定処理

次に、ステップ 1 において求めた部分共通交線と隣接する α 面の実3角形を求める。図 8(1) の例では、部分共通交線 FI と隣接する実3角形として ABC と BHI が求められる。そして、この2つの実3角形が形成する面 (α 面) と形状 β の実3角形面 (図 6(2) では3角形 A'B'C') との間で「面交差判定処理」(後述)を行う。交差しないと判定された場合は、この部分共通交線 FI は消去される。交差すると判定された場合はそのまま交線として残される。これらの処理をすべての部分共通交線に関して行う。すなわち、図 8(1) の例では、部分共通交線 IK, KG と隣接する実3角

形としてそれぞれ、ABC と IJK, ABC と KLC が求められる。そして、面交差判定処理が行われる。また図 8(2) の例では、部分共通交線 FI, IK, KG と隣接する実3角形としてそれぞれ、ABC と BHI, ABC と IJK, ABC と KLC が求められ、順次面交差判定処理が行われる。

(ステップ 3) 実3角形分割処理

形状 β の実3角形に対しては、前述の 3.2.2 項「交差する場合」と同じ処理を行い、共通交線と一致するエッジを生成する (図 7 参照)。そして、さらに位相分割を繰り返し、形状 α 側のすべての部分共通交線と一致するエッジを生成する。たとえば、図 6(2) の例において、形状 α 側の3角形 ABC の周辺の状況が図 8(1) であったとすると、形状 β 側の3角形 A'B'C' は図 8(3) のように、ゼロ3角形を用いて位相分割され、形状 α 側とまったく一致する部分共通交線が生成される。

(ステップ 4) 接続エッジ生成処理

α 面において、共通交線の端点と一致する頂点が存在しない場合は、位相分割を行う。図 8(1) の例では、共通交線の端点 F, G と一致する頂点 B, C が存在するので、位相分割は必要ない。しかし、図 8(2) の例では共通交線の端点 F および G と一致する頂点がない。そこで、図 8(2') のように位相分割を行い、頂点 F, G を生成する。

上記の処理により、 α 面において、部分共通交線の端点と一致する頂点は必ず存在することになる。しかし、部分共通交線と一致するエッジが必ず存在するとは限らない。そこで、一致するエッジが存在しない場合は、「位相変形」を行い、一致するエッジを生成する。位相変形に関しては文献 14) に詳しく説明してあるので省く。たとえば、図 8(2') の例では、部分共通交線 FI と一致するエッジは存在しない。そこで、ゼロ3角形 BIK と BKF に対して位相変形を行い、図 8(2'') のようにゼロ3角形 BIF と FIK を生成し、エッジ FI を生成する。

このように、部分共通交線の端点と一致する頂点が存在しない場合は位相分割を、部分共通交線と一致するエッジが存在しない場合は位相変形を行う。そして、このようにして生成した、部分共通交線と一致する両形状のエッジを接続する (局所的境界面接続を行う)。

3.2.4 エッジにおいて接する場合

図 6(3) に示すように、両形状の実3角形が互いにエッジにおいて接する場合である。この場合は以下のステップ 1~3 の処理を行う。

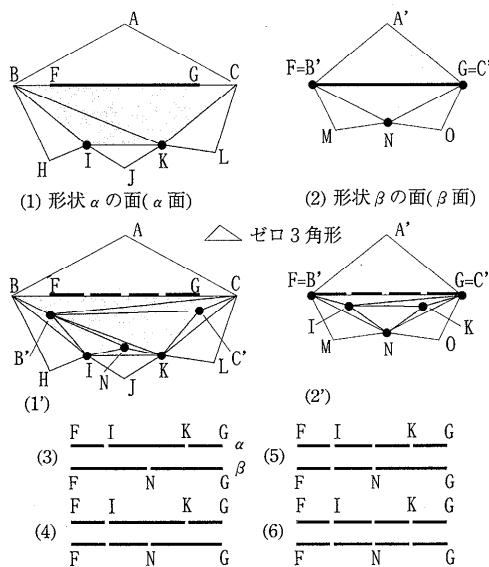


図9 共通交線の分割処理

Fig. 9 Cutting process of a common intersectional line.

(ステップ1) 共通交線分割処理

3.2.3 項「3 三角形面上となる場合」のステップ1と同様の処理により、まず、 α 面において、共通交線から部分共通交線（以下、これを α 部分共通交線と呼ぶ）を求める。そして、 β 面においても同様に部分共通交線（以下、これを β 部分共通交線と呼ぶ）を求める。たとえば、図9(1), (2)に示すように、両形状の3 三角形ABCとA'B'C'が共通交線をFGとして接しているとする、 α 部分共通交線および β 部分共通交線は図9(3)に示すようになる。

次に、両者の部分共通交線の長さを順次比較し、長い方の部分共通交線を分割し短い方にあわせる。長さが同一の場合すなわち両端点が一致する場合はそのままとなる。図9(3)の例では、 α 部分共通交線FIおよび β 部分共通交線FNを比較し、FNの方が長いのでFIとINに分割する（図9(4)）。(3)～(6)に示すように、両部分共通交線の両端点が一致となるように、分割処理を順次繰り返す。この処理により、両形状のすべての部分共通交線の端点は一致するようになる（図9(6)）。

(ステップ2) 面交差判定処理

ステップ2では、部分共通交線に隣接する実3 三角形のペアを、 α 面および β 面においてそれぞれ求める。図9の例では、部分共通交線FIに隣接する実3 三角形として、 α 面ではABCとBHI（図9(1'））を、 β 面ではA'B'C'とB'MN（図9(2'））をそれぞれ求める。そして、このそれぞれの2つの実3 三角形が形成する面

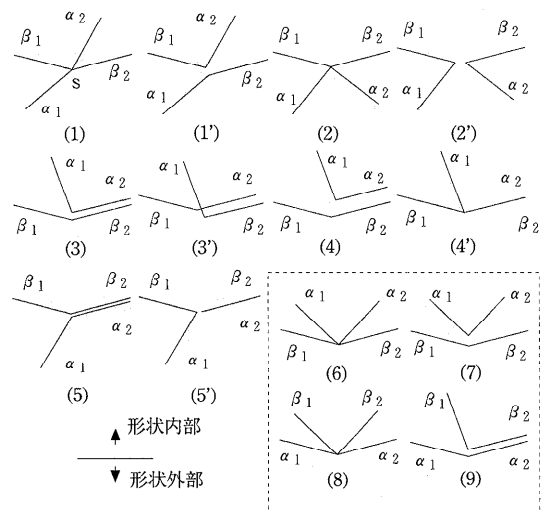


図10 面交差のパターン

Fig. 10 Intersectional patterns of two faces.

の間で、面交差判定処理（後述）を行う。交差しないと判定された場合は部分共通交線を消去する。面交差すると判定された場合はそのまま残す。以上の処理をすべての部分共通交線に対して行う。

(ステップ3) 接続エッジ生成処理

ステップ3の処理は3.2.3 項「3 三角形面上となる場合」のステップ4の処理とほとんど同様の処理となる。すなわち、上記のステップ2までの処理において生成された部分共通交線と一致するエッジを、形状 α および形状 β において、位相分割および位相変形によりそれぞれ生成する（図9(1'）と(2'））。もちろん、すでに部分共通交線と一致するエッジが存在している場合はこれらの処理は必要ない。そして、この一致する両形状のエッジを接続する。

3.3 面交差判定処理

面交差判定処理とは、形状 α と形状 β の境界面が局所において交差しているかどうかを判定する処理である。ここでは、両形状における、部分共通交線に隣接する両側の実3 三角形の位置関係により判定を行う。すなわち、図10(1)では、部分共通交線を s として、 s に隣接する形状 α の実3 三角形は α_1, α_2 であり、 s に隣接する形状 β の実3 三角形は β_1, β_2 である。そして、この α_1, α_2 が形成する面（ α 面）と β_1, β_2 が形成する面（ β 面）との間で面交差判定を行う。なお、図10は境界面の断面図である。

図10(1)～(5)に、面が交差していると判定するケースをまとめておく。(1)の場合は両形状の面が完全に交差する場合である。この場合は(1')に示すように、 α_1 と β_2, α_2 と β_1 を接続する。(2)の場合は両形状

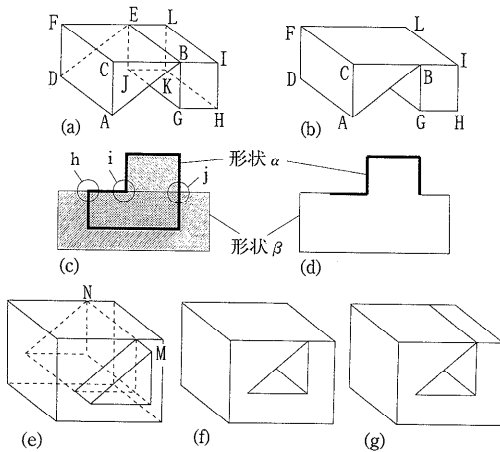


図 11 形状演算におけるルール

Fig. 11 Rules of Boolean shape operations.

の面が互いに他の形状の外側となり、かつ1つのエッジにおいて接する場合である。この場合は(2')のように接続する。これを交差すると判定するのは、本論文で提案している形状演算では、「エッジにおいて接する面は接続される」(条件1)という条件を設けているからである。すなわち、図11(a)の例では3角柱(形状 α)と4角柱(形状 β)がエッジBEにおいて接している。この両者を形状和すると、図11(b)のように、面CBEFとBILE、および面ABEDとGBEJは接続されて一体化される。

図10(3)の場合は、形状 α の面が形状 β の内側にある状態から重なる状態に変化する場合である。この場合も交差すると判定する。これも、「両面が法線ベクトルの向きを同じくして重なる場合は形状 α の面を残す」(条件2)という条件を設けているからである。すなわち、図11(c)の例に示す形状和演算を行うと、その結果は図11(d)となる。このように面が重なる部分は α 面が優先され残される。図11(c)において、hの部分がこの(3)のパターンとなっている。i, jの部分はそれぞれ、(9)(交差しない)、(1)(交差する)のパターンである。

一方、図10(6)~(9)の場合は、いずれも交差しないと判定する。(7)の場合は、(2)の場合の逆の場合である。この場合は、上記の条件1により、交差しないと判定する。すなわち、図11(e)に例示するように、直方体から3角柱を形状差する場合、直方体の上面は図11(f)に示すように一体面のままである。図11(g)に示すように上面にスリットは生成されない。(9)は(3)とは逆の場合である(図11(c)のiの部分)。この場合は条件2により、 β 面は残されないのだから交差にはならない。図10には図示していないが、両面が完全

に重なる場合も交差しないと判定する。

4. 計算機実験と考察

今回提案したデータ構造である「トライアングルエッジ」、およびこれを用いた新しい形状演算アルゴリズム(以下、NEWと呼ぶ)を評価するために、前回提案した交線を用いた形状演算アルゴリズム¹⁵⁾(以下、OLDと呼ぶ)と比較評価した。ここでは、比較対象形状例として以下の4形状を取り上げて、その処理時間(経過時間)を測定した。

- (1) 部品 — プリミティブ18個の和差により形状を生成し(図12(1))、さらに半分に切断する処理(図12(2))。
- (2) ブロック — 大きな薄板上に大きささまざまな1,000個の直方体を適当に配置し形状和したもの(図12(4))。
- (3) トーラス — 8個のトーラスの形状和(図12(5))。
- (4) 格子 — $5 \times 5 \times 3 = 75$ 個, $8 \times 8 \times 3 = 192$ 個, および $10 \times 10 \times 3 = 300$ 個の直方体の形状和(図12(6))。

なお、図12の表示画像はすべて、超3角形BRepから隠線処理した結果を表示したものである。図12(3)は3角形の辺も表示している。計算機は、IBM PC360 (Pentium Pro150 MHz, 128 Mbytes memory, WindowsNT3.51)を使用し、プログラミング環境として、マイクロソフト Visual C++を用いた。

OLDとNEWの処理時間に関する比較結果は表1となった。この表でエッジデータの項目以外における数字は、上段:OLDの処理時間、中段:NEWの処理時間、下段:NEWの処理時間をOLDの処理時間で割った比率である。エッジデータの項目に関しては、上段:総エッジデータ数(形状演算の過程で生成されたすべてのエッジデータの総和、消去されたデータも含む)、中段:接続エッジデータ数(境界面の接続に用いられたエッジデータの総数、カッコ内の数字は接続エッジデータ数を総エッジデータ数で割った比率、これを接続エッジ比率と呼ぶ)、下段:最終エッジデータ数(すべての形状演算を終了した時点でのエッジデータの数)である。また、分割接続処理の項目におけるOLDの数字は分割処理と接続処理の処理時間を加算したものである。

共通空間処理では、OLDはFデータ構造により表現されたすべての3角形面データにアクセスする。一方、NEWはEデータ構造により表現されたすべてのエッジデータにアクセスする。そして、これらのデータアクセス以外の処理はまったく同じ処理となる。したがって、両者の処理時間の差は、アクセスするデー

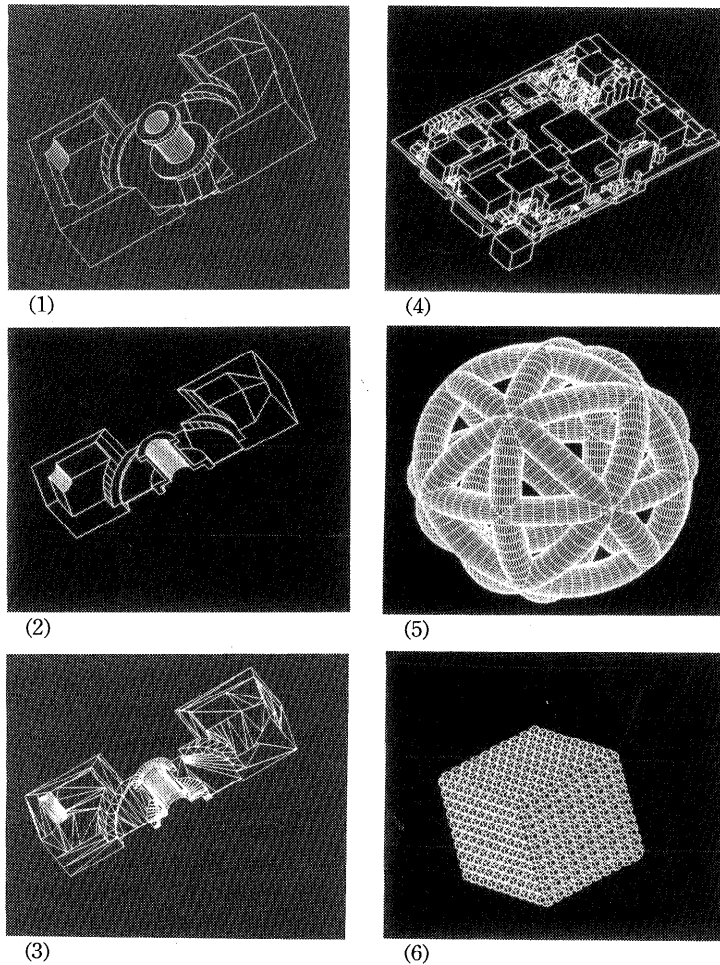


図 12 アルゴリズムの評価に用いた形状
Fig. 12 Shapes used for evaluation of algorithms.

表 1 実験結果
Table 1 Experimental results.

(単位: 秒, ただしエッジデータは個)

形状	処理	共通	分割接続	消去	統合	総時間	エッジデータ数
部品 (図12(2))		0.02	1.24	0.03	0.02	1.31	34,164
		0.01	1.30	0.02	0.00	1.33	9,432(0.276)
		(0.5)	(1.05)	(0.67)	(0.0)	(1.02)	3,126
ブロック (図12(4))		17.01	7.23	10.91	0.26	35.41	414,072
		24.31	6.69	5.12	0.29	36.41	115,544(0.279)
		(1.45)	(0.93)	(0.45)	(1.12)	(1.03)	52,992
トーラス (図12(5))		0.28	895.09	0.35	0.14	895.86	275,172
		0.40	918.04	0.39	0.36	919.19	41,020(0.149)
		(1.43)	(1.03)	(1.12)	(2.57)	(1.03)	118,272
格子5 (5×5×3)		0.17	4.36	0.13	0.12	4.78	168,384
		0.21	2.35	0.03	0.16	2.75	57,228(0.339)
		(1.23)	(0.54)	(0.23)	(1.33)	(0.58)	11,700
格子8 (8×8×3)		2.08	36.99	1.30	1.00	41.37	1,003,596
		2.92	18.86	0.68	0.92	23.38	340,420(0.339)
		(1.40)	(0.51)	(0.52)	(0.92)	(0.57)	43,776
格子10 (図12(6))		5.88	113.87	3.78	3.34	126.87	2,429,916
		8.99	61.22	2.24	2.56	75.01	822,372(0.338)
		(1.53)	(0.54)	(0.59)	(0.77)	(0.59)	82,800

タ数が直接的に反映される。すなわち、トライアングルエッジはFデータ構造に比べて3倍データ数が多くなるので(ここでは頂点データは除く)、共通空間処理のように単純に全データにアクセスをする処理はNEWの方が効率が悪くなる。

分割接続処理では、接続エッジ比率0.27あたりを境にして、それ以下ではNEWとOLDの処理時間は同程度、それ以上では、NEWの方がかなり効率が良くなる。これは、交差部分が増えると、OLDでは、付加的な交線データに関連する処理が増えること、かつ分割処理と接続処理が分離されていることによる効率の悪さに起因している。総処理時間に関しても、NEWは、接続エッジ比率が0.3程度より小さい場合は、OLDと同程度の処理時間である。しかし、接続エッジ比率が0.3を超えると数割程度速くなる。

5. おわりに

BRepにおけるデータ構造を考察し、超3角形BRepに適したEdge-basedデータ構造である、究極的に単純な「トライアングルエッジ」データ構造を提案した。そして、このデータ構造を用いた新しい形状演算のアルゴリズムもあわせて提案した。すなわち、従来のアルゴリズムでは別処理であった分割処理と接続処理を融合した、かつ交線データを不要とする単純化・簡略化された新しい形状演算アルゴリズムを提案した。

そして、今回取り上げた形状例では、従来の形状演算アルゴリズムに比べて、処理速度が最悪でも同程度、交差する部分が多い場合は数割程度高速化されることを計算機実験により実証した。

しかしながら、今回は非常に数少ない特定の形状例におけるアルゴリズムの評価であった。そこで今後の課題として、より複雑で実用的な形状を数多く用いて、今回提案したアルゴリズムを総合的に評価する必要がある。

トライアングルエッジは究極的に単純なデータ構造であり、3角形処理の持つ単純性が最大限生かされている。そして、これを用いた今回提案した形状演算アルゴリズムも簡略なものとなっており、信頼性の高い処理系を実現しやすい。しかしながら、その信頼性(演算誤差など)に関しても十分な対応および考察を行う必要がある。

参 考 文 献

- 1) Baer, A., Eastman, C. and Henrion, M.: Geometric modelling: A survey, *Computer-Aided Design*, Vol.11, No.5, pp.253-272 (1979).

- 2) Baumgart, B.G.: Winged Edge Polyhedron Representation, Rep.No.STAN-CS-72-320, Stanford Univ. (1972).
- 3) Baumgart, B.G.: Geometric Modeling for Computer Vision, Rep.No.STAN-CS-74-463, Stanford Univ. (1974).
- 4) Hubbard, P.M.: Constructive Solid Geometry for Triangulated Polyhedra, Tech. Rep. No. CS-90-07, Brown Univ. (1990).
- 5) Higashi, M., et al.: Face-Based Data Structure and its Application to Robust Geometric Modeling, *Third Symposium on Solid Modeling and Applications*, pp.235-246 (1995).
- 6) Laidlaw, D.H., Trumbore, W.B. and Hughes, J.F.: Constructive Solid Geometry for Polyhedral Objects, *Proc. SIGGRAPH'86*, Vol.20, No.4, pp.161-170 (1986).
- 7) Requicha, A.A.G.: Representations for Rigid Solids: Theory, Methods, and Systems, *ACM Computing Surveys*, Vol.12, No.4, pp.437-464 (1980).
- 8) Requicha, A.A.G. and Voelcker, H.B.: Solid Modeling: A Historical Summary and Contemporary Assessment, *IEEE Computer Graphics and Applications*, Vol.2, No.2, pp.9-24 (1982).
- 9) Requicha, A.A.G. and Voelcker, H.B.: Solid Modeling: Current Status and Research Directions, *IEEE Computer Graphics and Applications*, Vol.3, No.7, pp.25-37 (1983).
- 10) Weiler, K.: Edge-Based Data Structures for Solid Modeling in Curved-Surface Environments, *IEEE Computer Graphics and Applications*, Vol.5, No.1, pp.21-40 (1985).
- 11) Yamaguchi, F. and Tokieda, T.: Bridge Edge and Triangulation Approach in Solid Modeling, *Proc. Computer Graphics Tokyo 84*, pp.44-65 (1984).
- 12) Yamaguchi, F. and Tokieda, T.: A Unified Algorithm for Boolean Shape Operations, *IEEE Computer Graphics and Applications*, Vol.4, No.6, pp.24-37 (1984).
- 13) Yamaguchi, F.: A Unified Approach to Interference Problems Using a Triangle Processor, *Proc. SIGGRAPH'85*, Vol.19, No.3, pp.141-149 (1985).
- 14) 荒川佳樹: 面積ゼロ3角形を用いた3角形BRep, *情報処理学会論文誌*, Vol.36, No.2, pp.362-373 (1995).
- 15) 荒川佳樹: 超3角形BRepにおける高速形状演算アルゴリズム, *情報処理学会論文誌*, Vol.37, No.4, pp.624-634 (1996).
- 16) 新関雅俊, 小西史和, 吉田 誠, 山口富士夫: データ構造およびオイラーオペレータの双対性, *精密工学会誌*, Vol.59, No.8, pp.1233-1238 (1993).

- 17) 山口富士夫：4次元理論による図形・形状処理工学，日刊工業新聞社（1996）.

（平成9年1月10日受付）

（平成9年11月5日採録）



荒川 佳樹（正会員）

1954年生。1978年早稲田大学理工学部工業経営学科卒業。1980年同大学院理工学研究科修士課程機械工学専攻修了。同年松下電器産業（株）入社。1990年郵政省通信総合研究所入所。現在，同研究所通信システム部非常時通信研究室に勤務。主任研究官。超3角形/超4面体による形状モデリング，図形・画像処理の研究に従事。電子情報通信学会，精密工学会各会員。



山口富士夫（正会員）

1935年生。1959年早稲田大学第一理工学部機械工学専攻卒業。同年横河電機製作所入社。1967年機械振興協会技術研究所入所。コンピュータグラフィックスと曲面の自動設計に関する研究に従事。1978年九州芸術工科大学工業設計学科助教授。1979～1980年米国ユタ大学コンピュータサイエンス学科客員準教授。1986年早稲田大学理工学部機械工学科教授。著書「4次元理論による図形・形状処理工学」（日刊工業新聞社）他。精密工学会会員。