多面体の頑健な切断演算アルゴリズム

乾 正知† 寺門宏明†

多面体を任意の平面で切断する処理は，立体モデリングにおける基本的な演算の1つである．立体モデリングでは，幾何的な諸量の算出や判定を，浮動小数点演算を用いて行う．浮動小数点演算では数値誤差の発生が不可避なため，得られた結果に矛盾が生じやすく，最悪の場合処理が破綻してしまう．矛盾の多くは，近接した幾何要素の一致判定の失敗が原因で生じる．そこである微小な公差値よりも接近している要素を一致していると見なし，矛盾の発生を防止することが経験的に行われている．しかし従来の手法では，公差の導入が新たな矛盾の原因となる場合が多く，問題の解決になっていなかった．本研究では，既存の切断演算アルゴリズムを見直し，処理が破綻する原因を明らかにするとともに，処理を安定に行うための位相的な条件を導出した．次に必要な幾何的判定の手続きを，この条件を満たすように，切断平面の公差域に基づいて整合的に定義した．その結果，多面体の切断演算を安定に実行するアルゴリズムを実現したので報告する．実際にプログラムを作成し，数値実験により手法の有効性を確認した．

A Robust Algorithm for Slicing Polyhedra

MASATOMO INUI† and HIROAKI TERAKADO†

Polyhedron slicing with a plane is a fundamental operation in solid modeling. Numerical data describing the solid geometry are usually given using floating point numbers. Since floating point numbers are basically approximations, there may be imprecision that leads to inconsistencies about the represented object. Most programmers in practice are aware of this imprecision and give a small tolerance value for relaxing geometric tests. It is difficult to control all the implications of the approximate test, therefore two computations often yield contradictory results although they ask the same geometric questions. In this paper, the authors propose a new tolerance based algorithm for stable slicing of polyhedra. A topological condition for computing the sliced object is derived based on a classification of intersection points between polyhedron edges and the slicing plane. Tolerance based incidence tests are consistently designed so that the test results always satisfy the condition. Proposed algorithm is implemented and some computational experiments are demonstrated.

1. はじめに

多面体を任意の平面で切断する処理は，立体の断面図の作成やラピッドプロトタイピング用データの作成，BSP木のようなグラフィックスの高速処理用データの作成などで多用される，立体モデリングの基本的な演算の1つである[6]．立体モデリングでは，幾何的な諸量の算出や判定を，浮動小数点演算を用いて行う．浮動小数点演算では数値誤差の発生が不可避なため，演算結果に矛盾が生じやすく，最悪の場合処理が破綻してしまう．矛盾の多くは，近接した幾何要素の一致判定の失敗が原因で生じる．そこである微小な公差値よりも接近している要素を一致していると見なし，矛盾の発生を防止することが経験的に行われている．機械製品のCADでは，多面体のいくつかの頂点を通過するように切断平面を与える場合が多い．公差を用いた判定では，このような意図的に指示された交差を，数値誤差の影響を受けずに確実に判定できるので都合がよい．しかし従来の手法では，公差の導入が新たな矛盾の原因となる場合が多く，問題の解決になっていなかった[3),4)]．

公差の定義や一致判定の方法を改良し，矛盾を生じにくくした研究はこれまでにもいくつか知られているが，実用的な処理での利用には問題が多い．Segalは，一致と判定されるたびに公差を大きくすることで，安定な図形処理が実現できることを示している[8)]．この手法では，処理が進むにつれて誤差の許容量が拡大するため，得られる図形の精度が保証できない．Milenkovicは，2つの幾何要素が公差に基づいて一致

† 茨城大学工学部
Faculty of Engineering, Ibaraki University

と判定されると，それらを移動させ座標値などを完全に一致させることで，矛盾を解消する手法を提案している[7]．この手法では，移動した幾何要素と他の要素がさらに一致する可能性があり，処理が連鎖し図形の誤差が拡大してしまう．

計算中に発生する誤差の上限を見積もり，完全に信頼できる判定のみを利用して処理を進める方法も報告されているが（たとえば文献 2)），このようなアルゴリズムは複雑になりやすく，設計が困難である．アルゴリズムを見直し，幾何的判定の重複を取り除くことで矛盾の発生を防止する試みもあるが，同様の理由から実際の問題解決での利用は難しい[5),14)]．

杉原は，幾何的な判定の結果よりも，図形処理を進める前提となる位相的な条件の充足を優先することで，安定な切断処理を実現するアルゴリズムを提案している[10),11)]．しかしこのアルゴリズムは，現時点では凸多面体しか処理できない．またこの手法は，切断平面が立体の頂点を通過するような退化した交差が扱えないので，CAD のためのプログラム作成では利用が難しい．

多面体の切断処理では，必要な幾何的判定は，いくつかの平面の方程式の解法に帰着できるので，有理数演算を用いることで厳密に実現できる．有理数演算では，処理が進むにつれて分子と分母の桁数が増すため，演算時間と記憶容量が膨大なものになってしまう[1)]．杉原は，平面の方程式の係数を，数平面をある精度で分割した格子上の点に制限することで，多面体の集合演算を厳密かつ効率的に行う手法を開発した[9)]．この手法は，多面体の切断処理にも利用可能である．これらの手法は，立体や切断平面を配置する際の入力誤差も厳密に評価する．したがって，退化した交差が生じるように図形を配置しても，演算により得られた図形に，誤差に起因する意図しない微小構造ができてしまう[9)]．

本研究では，既存の多面体の切断演算アルゴリズムを見直し，処理が破綻する原因を明らかにするとともに，処理を安定に行うための位相的な条件を導出した．次に必要な幾何的判定の手続きを，この条件を満たすように，切断平面の公差域に基づいて整合的に定義した．その結果，任意の多面体の切断演算を安定に実行するアルゴリズムを実現したので報告する．この手法は，多面体の辺や頂点と切断平面の交差を判定する際に公差を用いる．したがって多面体や切断平面の位置姿勢に微小な誤差が生じても，意図的に指示された退化した交差を確実に判定できる．従来の手法とは異なり，処理中に公差の大きさは一定に保たれるので，要求精度の厳しい処理でも問題を生じない．この手法に基づいて実際にプログラムを作成し，数値実験により有効性を確認した．

2 章では，退化した交差が生じない場合の，多面体の切断演算アルゴリズムを概説する．3 章では，説明したアルゴリズムに基づいて，切断処理を安定に実行するための位相的な条件を導出する．4 章では，処理に必要な幾何的判定の手続きを，退化した交差が扱え，しかも安定な処理のための位相的な条件を満たすように定義しなおす．その際，退化した交差を確実に判定できるように，切断平面に公差域を導入する．5 章では，数値実験の方法とその結果について述べる．

2. 多面体の切断演算アルゴリズム

多面体 $\mathcal{P}$ を平面 s で分割し，s の上側の部分 $\mathcal{P}_a$ と下側の部分 $\mathcal{P}_b$ を得る処理を，多面体の切断演算とよぶ．下側の部分立体 $\mathcal{P}_b$ を得ることができるならば，s を裏返した面を用いて同じ計算を繰り返すことで，s の上側の部分立体 $\mathcal{P}_a$ を得ることができる．そこで以後は，$\mathcal{P}_b$ を得る処理のみを考える．

多面体は境界表現法により定義されており，幾何的な情報として，各頂点の座標値が与えられているものとする[6)]．多面体の各面の方程式が必要なときには，面の周囲の頂点の座標値に，文献 12) に紹介されている Newell の手法を適用し計算する．議論を簡単にするために，$\mathcal{P}$ の表面を覆う多角形はすべて連結しているものとし，$\mathcal{P}$ 内部に空洞などが生じていないことを仮定する．また $\mathcal{P}$ と s の間に退化した交差が生じていないこと，すなわち s が $\mathcal{P}$ の頂点を通過しないことを仮定する．最初の仮定は，処理の安定性に影響を与えない．2 つめの仮定については，後でこの制限を緩め，退化した交差も扱えるように手続きを拡張する．

2.1 処理の概略

以下に処理の概略を示す（**図 1** 参照）．このアルゴリズムは文献 13) に基づく．

Step 1 辺の分割 多面体 $\mathcal{P}$ の各辺と切断平面 s の交点を算出し，s と交差する辺を分割する．

Step 2 面の分割 s と交差する $\mathcal{P}$ の各面について，交点を新しい辺で接続し面を分割する．

Step 3 不要な辺の削除 $\mathcal{P}$ の辺のうち，s の上側に存在する辺を削除し，目的の部分立体 $\mathcal{P}_b$ を得る．

以下に各ステップの詳細を示す．

2.2 Step 1 辺の分割

多面体 $\mathcal{P}$ の各辺について，辺の両端の頂点が，切断平面 s のどちら側に存在するかを調べる．2 頂点が s の上下異なる側に存在するときには，辺と s の交点

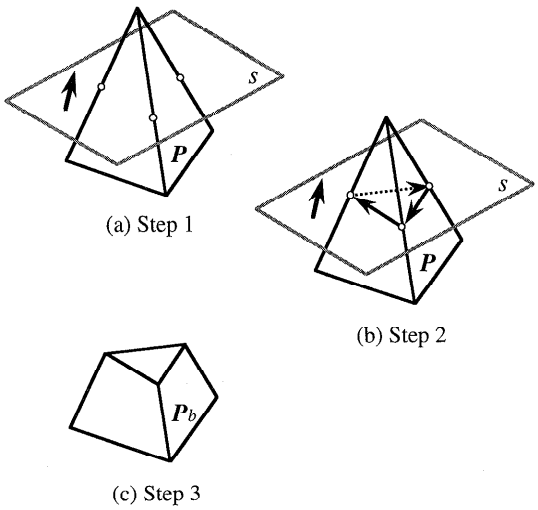


図1 多面体の切断演算アルゴリズムの概要
Fig. 1 Outline of a polyhedron slicing algorithm.

を計算し，その位置に新しい頂点を挿入することで辺を分割する．このような切断平面上の頂点を，交差頂点とよぶ．

2.3 Step 2 面の分割

前段の処理の結果，切断平面と交差する面の境界上には，交差頂点が生成されている．そこでこれらを結ぶ辺を生成し，各面を切断平面の上側と下側の部分に分割する．この処理は，さらに2つの部分に分けられる．

2.3.1 Step 2.1 交差頂点の分類

面 f と切断半面 s の間に交差が生じている場合を考える．f の境界は，ループとよばれる複数の閉曲線からなる．f の境界を構成する m 個のループを，$l_0, l_1, ..., l_{m-1}$ とする．各ループ l_i を構成する n 個の頂点を反時計回りにたどり，遭遇した頂点を順に $v_{i_0}, v_{i_1}, ..., v_{i_{n-1}}$ と名付ける．ただし添字は n の剰余系とする．

仮定により，辺の分割前には s は $\mathcal{P}$ の頂点を通過していないので，ループ l_i における交差頂点 v_{i_j} の直前の頂点 $v_{i_{j-1}}$ と直後の頂点 $v_{i_{j+1}}$ は，s の上下異なる側に存在する．このことを利用して，v_{i_j} を以下に定義する out の交差頂点と in の交差頂点へ分類する．

out の交差頂点 v_{i_j} の直前の頂点 $v_{i_{j-1}}$ は s の下側にあり，直後の頂点 $v_{i_{j+1}}$ は s の上側にある場合．ループ l_i は，v_{i_j} において，切断平面の裏から表へ出るので，これを out の交差頂点とよぶ（図2(a)参照）．

in の交差頂点 逆に $v_{i_{j-1}}$ は s の上側にあり，$v_{i_{j+1}}$ は s の下側にある場合．ループ l_i は，v_{i_j} において，切断平面の表から裏へ入るので，これを in の交差頂点とよぶ（図2(b)参照）．

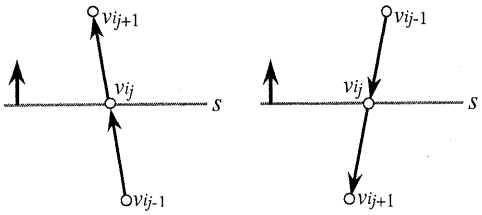
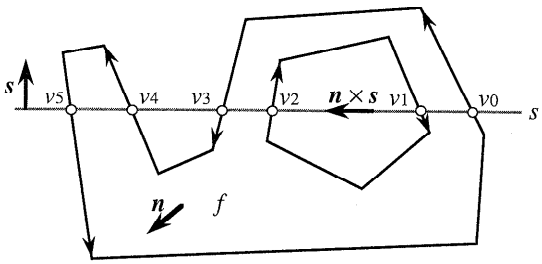


図2 交差頂点 v_{i_j} の分類
Fig. 2 Classification of intersection vertex v_{i_j}.



v0, v2, v4: "out" intersection vertices
v1, v3, v5: "in" intersection vertices

図3 面 f の境界上の交差頂点
Fig. 3 Intersection vertices on the boundary of face f.

図3の例では，v_0, v_2, v_4 が out の交差頂点，v_1, v_3, v_5 が in の交差頂点である．

2.3.2 Step 2.2 交差頂点の接続

f と s の交線上には，out の交差頂点と in の交差頂点が交互に並ぶ（図3参照）．そこで交差頂点を交線方向，すなわち f の法線ベクトル $\boldsymbol{n}$ と s の法線ベクトル $\boldsymbol{s}$ の外積 $\boldsymbol{n} \times \boldsymbol{s}$ 方向にしたがってソートする．具体的には，交線上にある基準点をとり，この点から交差頂点 v_{i_j} までの，$\boldsymbol{n} \times \boldsymbol{s}$ 方向を正とする符号付き距離 $D(v_{i_j})$ を測り，昇順にソートする．

次にソートされた順に，out の交差頂点と in の交差頂点を，それぞれを始点と終点とする新しい辺で接続する．定義から，付加された辺の右側が切断平面の上側に対応する．この交差頂点の接続作業は，以下の2つの場合に分類できる．

同一ループ上の交差頂点の接続 out の交差頂点と in の交差頂点が同一ループ上に存在する場合，これらの接続によって面 f は2つに分割され，新しい面 f' が生成される（図4(a)参照）．立体モデリングでは，これを mef 操作とよぶ．この操作にともない，f に属していたループが，その所属を

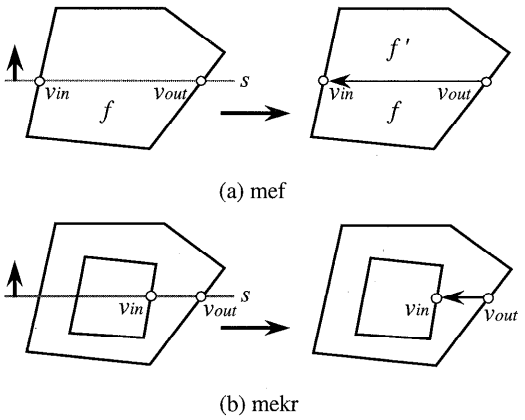


図4 交差頂点の辺による接続
Fig. 4 Connection of intersection vertices with an edge.

f' へ変える場合がある．具体的には，ループ l_i 上の点 p が新しく生成された面 f' の内部に含まれる場合には，l_i の所属を f から f' へ変更する．このループの移動は，切断処理の安定性には影響を与えない．

異なるループ上の交差頂点の接続 out の交差頂点と in の交差頂点が，同じ面の異なるループ上に存在する場合には，接続によってこれら2つのループが1つに併合される（図4(b)参照）．これを mekr 操作とよぶ．

図3の例では，まず v_0 と v_1 を mekr 操作で接続する．その結果残りの交差頂点は，すべて同一ループ上に存在することになるので，v_2 と v_3 および v_4 と v_5 を，ともに mef 操作で接続する．

2.4 Step 3 不要な辺の削除

交差頂点は，いずれも辺の分割により生成されたものなので，2つの面と隣接している．交差頂点 v が，面 f と f' に隣接している場合を考える．f と f' ではループ上の v をたどる順序が逆になるので，f において v が out の交差頂点と判定されたなら，同じ頂点は f' では必ず in の交差頂点と判定される．したがって Step 2 で付加された辺は，その右側が切断平面 s の上側となるように向き付けられたサイクルを構成する（図1(b)参照）．

s の上側の辺は，切断後の立体には不用である．そこで多面体 $\mathcal{P}$ の辺のうち，サイクルの右側の部分に含まれているものをすべて探索し削除する．その結果，s の下側に相当する部分立体 $\mathcal{P}_b$ が得られる．

3. 安定な切断処理のための位相的な条件

上述のアルゴリズムに基づくプログラムは，数値誤差の影響で処理に失敗してしまうことがある．本章では，そのような破綻の生じない，安定な処理を実現するための位相的な条件を示す．

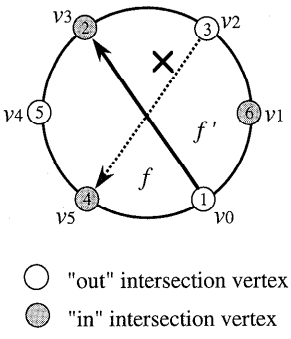


図5 交差頂点の接続に失敗するケース
Fig. 5 Possible failure in connecting intersection vertices.

3.1 破綻の原因

処理の破綻は，幾何的な判定結果と立体の位相構造が矛盾すると発生する．Step 1 の辺の分割では，そのような矛盾は生じない．また Step 2 で正しく辺が付加されれば，それらは必ずサイクルを構成するので，Step 3 の不要な辺の削除で問題が生じることはない．したがって処理が失敗し得るのは，Step 2 の交差頂点の接続処理だけである．その原因は以下の2種類に分類できる．

(1) 数値誤差が原因で，交線上の交差頂点の順序に逆転が生じ，out と in の頂点が交互に並ばない場合がある．その結果，out の交差頂点と接続すべき in の交差頂点が検出できず，処理に失敗してしまう．

(2) 交線上に out と in の交差頂点が交互に並んでいても，接続不可能な場合がある．**図5**に例を示す．図中，円は面のループを模式化したものであり，円周上の白い丸は out の交差頂点，また灰色の丸は in の交差頂点を表す．また丸の中の数字は，ソートにより得られた交線上での各交差頂点の順序を表している．ソートされた順に，1番目の out の頂点 v_0 と2番目の in の頂点 v_3 を mef 操作で接続した後（図中の太い矢印），3番目の out の頂点 v_2 と4番目の in の頂点 v_5 の接続を試みると，これらは異なる面に属しているため mef 操作も mekr 操作も適用できず，処理に失敗してしまう．

3.2 位相的な条件の導出

切断平面と交差する面の各ループ上に，out と in の交差頂点が交互に現れていれば，前述のアルゴリズムに修正を施すだけで，安定な切断演算が実現できる．

その理由を以下に示す．

3.2.1 交線上の交点順序の修正

閉じたループ上に交互に現れているのだから，面の境界上の out と in の交差頂点は同数である．そこで交線方向にしたがって交差頂点をソートする際に，out の交差頂点と in の交差頂点を別々にソートし，次にそれらを out と in の頂点が交互に現れるように併合すれば，最初の問題を回避できる．

例として面の境界上に，out と in の交差頂点が，それぞれ 2 個得られた場合を考える．out の頂点を v_0, v_1，in の頂点を v_2, v_3 とする．前述の手順に従い，交線上の基準点からこれらの頂点までの符号付き距離を測ったところ，$D(v_0) < D(v_1)$, $D(v_2) < D(v_3)$ であった．そこで，まず out の頂点群と in の頂点群を別々にソートし，2 つの列 $v_0 \to v_1$ と $v_2 \to v_3$ を得る．次にそれらを，out と in の頂点が交互に現れるように併合し，$v_0 \to v_2 \to v_1 \to v_3$ を得る．

この新しいソート手順を採用することで，数値誤差の有無にかかわらず，接続すべき交差頂点の組を確実に決定できる．

3.2.2 頂点の接続手順の修正

mef 操作は，面のループの分割と見なすことができる．逆に mekr 操作は，ループの併合と見なすことができる．分割/併合前のループ上に，out と in の交差頂点が交互に現れているならば，分割/併合後のループ上でもこれらは交互に現れる．**図 6** (a) には，mef 操作によるループの分割を模式的に示した．ループ上の交差頂点の列 $[v_0, v_1, v_2, v_3, v_4, v_5]$ は，v_1 と v_4 を mef 操作で接続すると，2 つのループ上の列 $[v_0, v_5]$ と $[v_2, v_3]$ へ変化するが，いずれの列でも out と in の交差頂点が交互に現れている．図 6 (b) には，mekr 操作によるループの併合を示した．こちらでも併合後のループ上に，out と in の交差頂点が交互に現れている．

3.1 節に示した 2 つめの問題は，接続を予定していた out と in の交差頂点の組が，mef 操作により異なる面に分けられてしまうことが原因である．上述のように，out と in の交差頂点がループ上に交互に現れていれば，mef 操作や mekr 操作を何回繰り返しても，その性質は失われない．したがって面の境界上の out と in の交差頂点は，つねに同数である．そこで mef 操作により面が分割されたときには，分割後の各面において，交差頂点を交線方向でソートしなおし，接続すべき out と in の頂点の組を更新する．この更新後の組に基づいて接続操作を実行すれば，処理の破綻を回避できる．

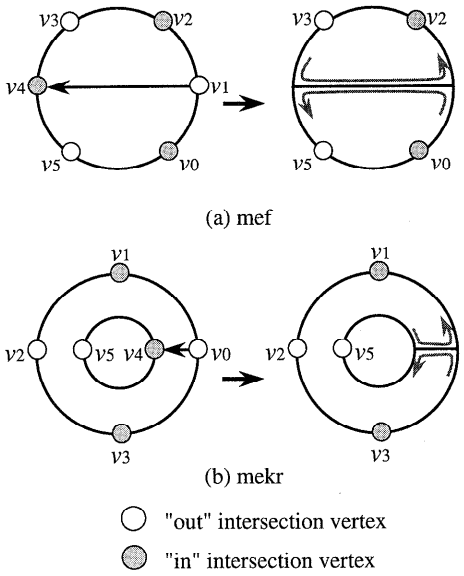


図 6 mef 操作や mekr 操作にともなうループの分割と併合
Fig. 6 Subdivision and merging of boundary loops according to mef and mekr operations respectively.

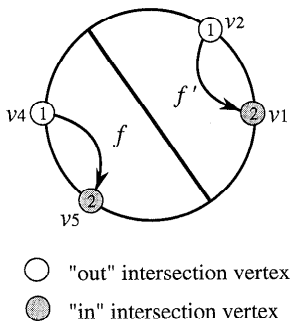


図 7 面の分割後の交差頂点の再ソートによる処理の破綻の回避
Fig. 7 Avoidance of the connection failure by re-sorting intersection vertices after a face subdivision.

図 5 の例では，mef 操作による面の分割後に，各面の境界上の交差頂点をソートしなおすと，面 f については $v_4 \to v_5$，面 f' については $v_2 \to v_1$ という新しい順序が得られる（**図 7** 参照）．ただしこのソートには，out と in の交差頂点が必ず交互に現れる新しい手法を用いる．その結果，今度は同一ループ上の v_4 と v_5 の組と v_2 と v_1 の組を，それぞれ mef 操作で接続することになるので，処理の破綻は生じない．

以上の議論から，「面のループ上に out と in の交差頂点が交互に現れること」が，多面体の切断演算を安定に行うための位相的な条件となる．

4. 退化した交差の扱い

次に，切断平面が多面体の頂点を通過する退化した交差も扱えるように，幾何的な判定手法を拡張する．

4.1 交差頂点の分類規則の拡張

退化した交差を考慮すると，ループにおける交差頂点の直前や直後の頂点も交差頂点の可能性があるため，Step 2.1 の分類規則は利用できない．そこで規則を以下のように拡張する．

out の交差頂点 交差頂点 v_{i_j} の直前の頂点 $v_{i_{j-1}}$ は切断平面 s の下側にあり，直後の頂点 $v_{i_{j+1}}$ は s 上もしくは s の上側にあるとき，v_{i_j} を out の交差頂点とよぶ．

in の交差頂点 逆に $v_{i_{j-1}}$ は s 上もしくは s の上側にあり，$v_{i_{j+1}}$ は s の下側にあるとき，v_{i_j} を in の交差頂点とよぶ．

図 8 (a) の例では，v_0 と v_4 が out の交差頂点，v_2 と v_5 が in の交差頂点と判定される．これら以外にも，v_1, v_3, v_6 など交差頂点の状態はいくつか考えられるが，これらはすべて out や in の頂点とは考えず，Step 2.2 以降の処理では無視する．図 8 (a) から明らかなように，この分類規則を採用すれば，面のループ上に out と in の交差頂点が，必ず交互に現れる．

4.2 不要な辺を削除するためのサイクルの構成

交差頂点の分類を上述のように拡張しても，それらを接続する辺は，その右側が切断平面 s の上側に対応するように向き付けられたサイクルを構成する．以下にその理由を示す．

交差頂点 v_{i_j} の，ループ上での直前の頂点 $v_{i_{j-1}}$ が s 上の場合には，v_{i_j} と $v_{i_{j-1}}$ を結ぶ辺をその中点で分割し，その点を s の上側へ（仮想的に）微小量移動する．同様に，v_{i_j} の直後の頂点 $v_{i_{j+1}}$ が s 上の場合には，v_{i_j} と $v_{i_{j+1}}$ を結ぶ辺をその中点で分割し，その点を s の上側へ（仮想的に）微小量移動する．図 8 (b) には，そのような辺の分割と移動を施した結果を示した．図形にこのような修正を加えても，out や in の交差頂点の判定を覆すことはない．またこの修正によって，新たに out や in の交差頂点が生成されることもない．

立体に修正を加えた結果，交差頂点のループ上での直前や直後の頂点は，必ず切断平面 s の上側もしくは下側に存在する．以後はこの修正済みの立体について考える．退化した交差を許した結果，交差頂点 v が 3 枚以上の面と隣接している可能性がある．**図 9** には，そのような交差頂点の近傍を，切断平面の法線方向から眺めた様子を示した．図中の各扇形が v に隣接する面を表す．v に接続する各辺の v ではない側の頂点には，その頂点が s の上下どちら側に存在するかを，above もしくは below の記号で示した．

v に隣接する各面 f_i のループを反時計回りにたどり，v の直前の頂点と v の直後の頂点が，s の上下どちら側に存在するか調べる．それらが below → above と移り変わるなら，f_i において v は out の交差頂点となる．逆に above → below と移り変わるなら，v は in の交差頂点となる．図 9 から明らかなように，v を out と判定する面（図 9 では f_3 と f_5）と，in と判定する面（f_0 と f_4）は必ず同数存在し，それらは v の周囲に交互に現れる．

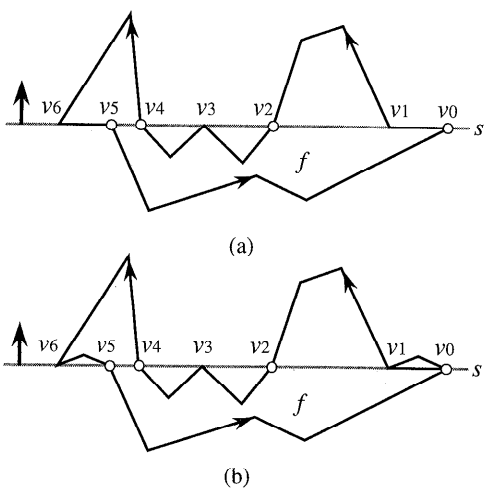


図 8 退化した交差を考慮した交差頂点 v_{i_j} の分類例
Fig. 8 Classification example of intersection vertices if degenerate intersections are allowed.

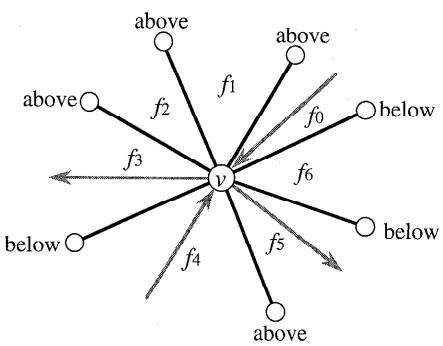


図 9 交差頂点 v に隣接する面の状態
Fig. 9 Faces adjacent to intersection vertex v.

交差頂点を辺で結ぶと，v を in と判定した面には v を終点とする辺が付加され，v を out と判定した面には v を始点とする辺が付加される（図 9 中の灰色の辺参照）．v の周囲には，これらの v を終点とする辺と始点とする辺が交互に現れる．そこで v を終点とする辺の次に，v を始点とする反時計回りに次の辺をたどるようにすれば，交差頂点を結ぶ辺は，その右側が s の上側に対応するように向き付けられたサイクルを必ず構成する．

4.3 退化した交差の確実な検出

CAD では，多面体のいくつかの頂点を通過する平面を，切断平面として用いることが多い．しかし数値誤差の影響で，入力者の意図する退化した交差が正しく検出されない場合がある．そこで，切断平面 s を中心とする厚さ 2ε の板状の公差域を考え，頂点 v_{i_j} が公差域に含まれている場合には「切断平面上」と判定し，公差域の上（下）側の半空間に含まれているときには，切断平面の「上（下）側」と判定する．out と in の交差頂点の分類規則を，公差域を利用するものに変更しても，ループ上に out と in の交差頂点が交互に現れるという条件は満たされる．したがって公差域を導入しても，前述の方法を用いることで，切断処理を安定に実現できる．

5. 数値実験

本手法に基づいて，C 言語で多面体の切断演算プログラムを作成し，数値実験により手法の有効性を評価した．このプログラムは，幾何計算や mef 操作などの位相的な処理を行う汎用的なパッケージと，350 行ほどの切断処理用プログラムからなる．計算には倍精度の浮動小数点演算を用いた．また公差の大きさは，$\varepsilon = 1 \times 10^{-3}$ とした．これは通常の公差値の約 1,000 倍に相当する．公差値を大きくすると幾何的な矛盾が生じやすいので[9]，以下の実験は通常よりもはるかに破綻しやすい状況で行われている．まず高さ 0.1 以下の微小かつランダムな凹凸を多数含む，19,734 枚の面からなる半径 100 の球状の多面体を生成した．次に半径 100 の球に接する 5,000 枚の平面をランダムに生成し，多面体の凸の部分をそぎ落とす実験を行った（**図 10** 参照）．**図 11** には処理後の多面体の状態を示した．この例以外にも，公差値や立体を変えて様々な数値実験を行ったが，実験した範囲では，処理が破綻する例を見つけることができなかった．したがって本手法は，実用的には十分に頑健と考えられる．

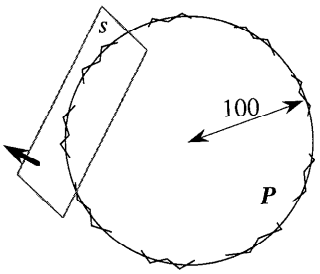


図 10 数値実験の方法
Fig. 10 Computational experiment method.

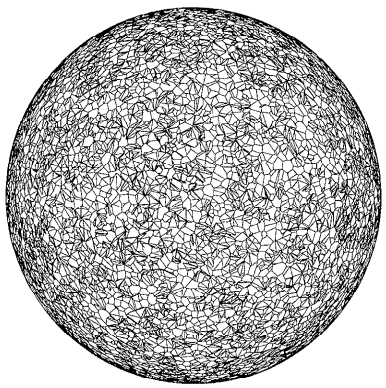


図 11 数値実験の結果
Fig. 11 Result of the computational experiment.

6. まとめ

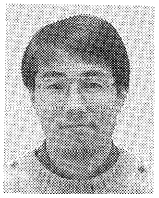
本論文では，退化した交差を含む多面体の切断演算を，安定に実行するアルゴリズムについて議論した．まず退化した交差が発生しないという仮定のもとで，既存の多面体の切断アルゴリズムの見直しを行った．その結果，切断平面と交差する面のループ上に，out と in の交差頂点が交互に現れていれば，アルゴリズムに修正を施すだけで，安定な切断演算が実現できることを示した．次に退化した交差の判定法を，上述の条件を満たすように，切断平面の公差域に基づいて整合的に定義した．

本手法では，多面体や切断平面の配置時に微小な誤差が生じても，意図的に指示された退化した交差を確実に判定できる．また処理中に公差の大きさは一定に保たれるので，要求精度の厳しい処理でも問題を生じない．提案したアルゴリズムに基づいてプログラムを作成し，数値実験によりその有効性を検証した．提案した手法は，より複雑な図形処理でも利用可能である．現在，我々は同様の手法に基づく多面体の頑健な集合演算アルゴリズムを検討している．

参考文献

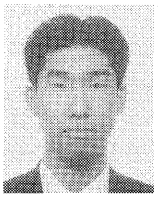
1) Benouamer, M., Michelucci, D. and Peroche, B.: Error-free Boundary Evaluation Using Lazy Rational Arithmetic – A Detailed Implementation, *Proc. 2nd ACM Symp. Solid Modeling and Applications*, pp.115–126 (1993).
2) Guibas, L., Salesin, D. and Stolfi, J.: Epsilon Geometry: Building Robust Algorithms from Imprecise Calculations, *Proc. 5th ACM Annual Symp. Computational Geometry*, pp.208–217 (1989).
3) Hoffmann, C.M., Hopcroft, J.E. and Karasick, M.S.: Towards Implementing Robust Geometric Computations, *Proc. 4th ACM Annual Symp. Computational Geometry*, pp.106–117 (1988).
4) Hoffmann, C.M.: *Geometric and Solid Modeling: An Introduction*, Morgan-Kaufmann (1989).
5) Hoffmann, C.M., Hopcroft, J.E. and Karasick, M.S.: Robust Set Operations on Polyhedral Solids, *IEEE Computer Graphics and Applications*, Vol.9, No.6, pp.50–59 (1989).
6) Mäntylä, M.: *An Introduction to Solid Modeling*, Computer Science Press (1988).
7) Milenkovic, V.: Verifiable Implementations of Geometric Algorithms Using Finite Precision Arithmetic, *Artif. Intell.*, Vol.37, pp.377–401 (1988).
8) Segal, M.: Using Tolerances to Guarantee Valid Polyhedral Modeling Results, *Computer Graphics*, Vol.24, No.4, pp.105–114 (1990).
9) 杉原厚吉，伊理正夫：計算誤差による暴走の心配のないソリッドモデラの提案，情報処理学会論文誌，Vol.28, pp.962–974 (1987).
10) Sugihara, K.: A Robust and Consistent Algorithm for Intersecting Convex Polyhedra, *Computer Graphics Forum*, Vol.13, No.3, pp.C45–C54 (1994).
11) 杉原厚吉：計算幾何工学，培風館 (1994).
12) Sutherland, I.E., Sproull, R.F. and Schumacker, R.A.: A Characterization of Ten Hidden-surface Algorithms, *ACM Comput. Surv.*, Vol.6, No.1, pp.1–55 (1974).
13) Vaněček, G.: Spatial Partitioning of a Polygon by a Plane, *Graphic Gems V*, Academic Press (1995).
14) Zhu, X., Fang, S. and Brüderlin, B.D.: Obtaining Robust Boolean Set Operations for Manifold Solids by Avoiding and Eliminating Redundancy, *Proc. 2nd ACM Symp. Solid Modeling and Applications*, pp.147–154 (1993).

(平成8年11月12日受付)
(平成9年6月3日採録)



乾　正知（正会員）

昭和36年生．昭和61年東京大学大学院工学系研究科情報工学専攻修士課程修了．平成5年より茨城大学工学部システム工学科助教授．工学博士．形状処理技術とその機械生産自動化への応用に興味を持つ．精密工学会，IEEE，ACM各会員．



寺門　宏明

昭和48年生．平成8年茨城大学工学部システム工学科卒業．現在同大学大学院システム工学専攻博士前期課程在学中．頑健な幾何処理アルゴリズムの設計法に興味を持つ．精密工学会会員．