数値的に安定な3次元Voronoi図構成算法とその応用

福 田 浩

(1)

数値的に安定な 3 次元 Voronoi 図構成算法とその応用

稲垣　宏

目 次

序章

背景

近年の計算機の高性能化に伴ない，計算の対象として数値や文字データに限らず図形や画像データまでも含まれるようになり，計算機の利用分野は飛躍的に拡大しつつある．

そのような時代背景を受けて，幾何的な構造を有する情報を計算機で高速に処理するアルゴリズムに関する研究が，計算幾何学という名の下に，この２０年ほどの間に極めて精力的に行なわれ，多くの成果を上げてきた．

幾何学自体ははるか古代から研究されてきた学問であるが，計算幾何学においてはそれをアルゴリズムの設計と解析という観点から統一的に捉えようとするアプローチがなされている．

実用的な面から見ても，大規模な幾何情報処理を必要とする分野は，VLSI のレイアウト設計，地図情報処理，パターン認識，コンピュータグラフィックス，ロボティックスなど広範囲に渡っており，多くの産業分野において効率的な幾何アルゴリズムの確立が求められている．

これまでの経緯

計算幾何学においては，非常に多くの高速アルゴリズムが提案されてきたが，これらのアルゴリズムが産業応用などで有効に利用されているとは必ずしもいえない．これは，理論的な研究の進歩が早いので，実用的なシステムを構築するのに今しばらく時間がかかるといった要因もあろう．しかし，それとは別に，“正確な計算ができると仮定された理論的な世界において提案されたアルゴリズムを，有限精度の値を扱う計算機上で走らせると，計算誤差のために理論通り正しく実行されるとは限らない” と

いったやっかいな問題を含んでいるのである．

計算幾何学におけるアルゴリズムの多くは幾何的構造を巧妙に利用して高速化を計っている．この種のアルゴリズムを計算機上で実行した場合，計算誤差の影響により途中で判断を誤ると，アルゴリズムの理論的基盤である幾何的構造そのものに矛盾が生じ，処理が破綻しやすい．その結果，無限ループに陥ったり，異常終了したり，意味のない解を出力したりする．

この問題は，単にアルゴリズムをプログラムに翻訳する際の間違いであるといったレベルの問題ではなく，理論と現実のギャップに根付く深刻な問題なのである．

最近になってこの問題の深刻さが次第に認識されるようになり [Hoffmann, 1989]，有限精度の計算を使っても安定して実行できる幾何アルゴリズムの設計手法がいくつか提案されている．代表的なものを列挙する．

方法 1 すべての計算をあらかじめ固定した整数格子上で近似的に行なう [Greene and Yao, 1986].

方法 2 位相構造を正しく判別できるだけの計算精度を確保する [Benouamer, Michelucci and Peroche, 1993; Karasick, Lieber and Nackman, 1991; Milenkovic, 1989; Ottmann, Thiemt and Ullrich, 1987; 杉原, 伊理, 1987; 吉田, 1986].

方法 3 誤差解析を行ない，数値計算結果の中で信頼できるものを選び出す [Dobkin and Silver, 1988; Fortune, 1989; Guibas, Salesin and Stolfi, 1989; Hoffmann, Hopcroft and Karasick, 1988; Milenkovic, 1988].

方法 4 位相構造に矛盾が生じたら，数値計算結果に合うように位相構造を修正する [Segal and Sequin, 1985].

方法 5 数値計算結果よりも位相構造の無矛盾性を優先させる [杉原, 1991; Sugihara and Iri, 1992; Sugihara, 1992]

本研究の位置付け

上記のアプローチのうち，[方法 1] では十分な解像度を得るためには膨大な記憶容量が必要になり，[方法 2] では計算コストが膨大になり，[方法 3] では複雑な誤差解析が必要になり，[方法 4] では矛盾を解消するための複雑な推論機構が必要になる．本研

究では [方法5] のアプローチを採る．そして，このアプローチを “位相優先法” と呼ぶことにする．位相優先法においては，計算誤差の存在を積極的に認め，そのような世界では “正確な数値判定をすることがそもそもできないのである” という立場をとる．

したがって，数値計算を絶対的なものとして信じることをやめ，計算誤差の影響を受けない組合せ計算を処理の中心におく．そのため，アルゴリズムの骨格を位相構造の変化として記述し，位相構造に矛盾をきたさない範囲で最も確からしい解を選ぶためにまたそのためだけに数値計算結果を利用する．

この位相優先法を適用した例として，平面 Voronoi 図構成問題 [伊理, 杉原, 1988; 杉原, 1991; 大石, 杉原, 1991]，2次元線分 Voronoi 図構成問題 [今井, 杉原, 1994]，ソリッドモデリング [Hoffmann, *et al.*, 1988]，凸多面体の切断問題 [Sugihara, 1994a]，3次元凸包問題 [Sugihara, 1994b] などがある．

本論文では，文献 [伊理, 杉原, 1988; 杉原, 1991; Sugihara, 1992; Sugihara and Iri, 1992; Sugihara and Iri, 1994] で2次元 Voronoi 図構成問題に対して提案された考え方を3次元へ拡張し，3次元 Voronoi 図を構成するための計算誤差に強いアルゴリズムを提案する．さらに，アルゴリズムの数値的安定性を利用したいくつかの応用例について述べる．

3次元の Voronoi 図は，空間内の補間，点パターンの解析，3次元有限要素法のためのメッシュ生成，3次元図形の薄面化，骨格線抽出，ロボットの衝突回避作業計画など多くの応用をもっている [Aurenhammer, 1991; Okabe, *et al.*, 1992; Dey, *et al.*, 1992; Bern and Eppstein, 1992; Schumaker, 1993; 名取, 野寺, 1989]．本研究の目的はそれらの応用に対して数値誤差による破綻の恐れがない安定したアルゴリズムを提供することにある．

なお，3次元 Voronoi 図においては，2次元 Voronoi 領域（多角形）の境界と異なり，Voronoi 領域（多面体）の境界に自然な順序がつかないなどの差があるため，ここで必要な位相的性質の保持は，2次元の場合の単純な延長ではなく，本質的に新しい工夫を含むものである．

本論文の全体の構成は次のようである．最初に Voronoi 図・Delaunay 図の定義と逐次添加型の Voronoi 図構成算法について概略を示す（第1章）．次に，3次元 Voronoi 図構成問題において，位相優先法の適用により数値的安定化が計られた新しい逐次添加型算法を提案し，計算機実験結果も併せて報告する（第2章）．そして，この計算機実験結果を検討中にわかった3次元 Delaunay 図特有の問題点について述べ，その対処

方法を構成する（第3章）．次に，位相優先法に基づいて設計された3次元 Voronoi 図構成算法は，その数値的安定性を積極的に利用することにより，わずかな変更で制約つきの3次元 Delaunay 図の近似構成法へ拡張できることを示す（第4章）．最後に，有限要素法における3次元メッシュの自動生成問題への応用について述べる（第5章）．

第 1 章

Voronoi 図とその構成算法

1.1 Voronoi 図

d次元空間の有限点集合 $P = \{p_1, p_2, \cdots, p_n\}$ に対して，$V(p_i) = \{x | x \in \mathbf{R}^d, l(x, p_i) < l(x, p_j), j = 1, 2, \cdots, n,\ j \neq i\}$ （ただし，$l(x, p_i)$ は，点 x と点 p_iのユークリッド距離を表す）を，点 p_iの Voronoi 領域 (Voronoi region) といい，Voronoi 領域 $V(p_1), V(p_2), \cdots, V(p_n)$ が定める d 次元空間の分割を，Pに対する Voronoi 図 (Voronoi diagram) とよび，Vor(P) で表す．このとき，Pの要素を母点 (generator) という．すなわち，Voronoi 領域 $V(p_i)$ は，与えられた他のどの点よりも点 p_iに近い点全体のなす領域であり，点 p_iの勢力圏とよばれることもある [Preparata and Shamos, 1985; 伊理, 腰塚, 他, 1993]．2 次元の場合の例を図 1.1に示す．黒丸が母点で，実線が Voronoi 図 である．

3 次元の場合には，Voronoi 領域 $V(p_i)$ は多面体となり，Voronoi 図 Vor(P) は 3 次元空間の多面体分割を与える．この多面体の頂点を Voronoi 点 (Voronoi point)，辺を Voronoi 辺 (Voronoi edge)，面を Voronoi 面 (Voronoi face) とよぶ．

Voronoi 図の性質と応用については，文献 [Aurenhammer, 1991], [Okabe, *et al.*, 1992] などに詳しい．

1.2 Delaunay 図

Voronoi 図は距離の概念に基づいた母点の勢力圏分布を表しているので，Voronoi 領域が境界を共有する 2 個の母点は “隣あっている” と見なすことができる．隣あっている母点同士を線分で結ぶことにより，Voronoi 図の双対図形ができるが，これを

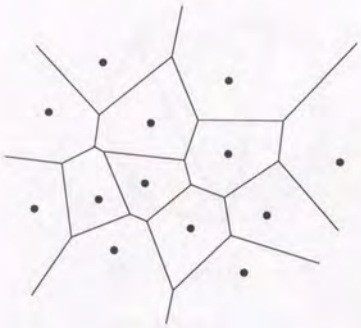


図 1.1. 2次元の Voronoi 図

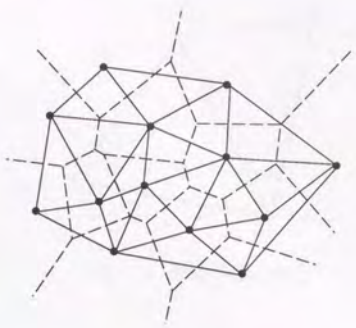


図 1.2. 2次元の Delaunay 図

Delaunay 図 (Delaunay diagram) とよび，Del(P) と表す．Del(P) の辺を Delaunay 辺 (Delaunay edge) という．

2次元の場合，Delaunay 図 Del(P) は，母点集合 P の凸包の内部を，母点を頂点とする多角形で分割する．この分割でできた多角形を Delaunay 多角形 (Delaunay polygon) とよぶ．特に Delaunay 多角形が三角形のとき，Delaunay 三角形 (Delaunay triangle) とよばれる．すべての Delaunay 多角形が三角形となるとき，母点集合は一般の位置にあるといわれる．図 1.1と同じ母点配置に対する Delaunay 図を図 1.2に示す．破線が Voronoi 図で，実線が Delaunay 図である．

3次元の場合には，母点集合 P の凸包の内部を，母点を頂点とする多面体で分割することになる．この分割でできた多面体領域を Delaunay 多面体 (Delaunay polyhedron) とよぶ．特に Delaunay 多面体が四面体のときには，Delaunay 四面体 (Delaunay tetrahedron) とよばれる．すべての Delaunay 多面体が四面体のとき，母点集合は一般の位置にあるという．

このような Voronoi 図，Delaunay 図といった図形は，与えられた点集合に対して "近さ" に基づいた構造を与えることができるため，地図情報処理，補間，点パターンの解析，有限要素法のためのメッシュ生成，骨格線抽出，ロボットの衝突回避作業計画など多くの応用をもっている [Preparata and Shamos, 1985; 伊理，腰塚，他, 1993; Aurenhammer, 1991; Okabe, *et al.*, 1992; Dey, *et al.*, 1992; Schumaker, 1993; 名取，野寺, 1989].

ここで， Delaunay 図の性質のうち，本論文に関連の深い空円性と最小角最大性と

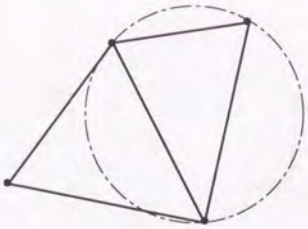


図 1.3. 空円性

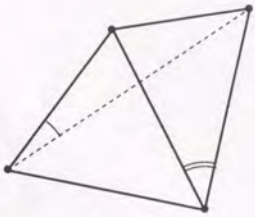


図 1.4. 最小角最大性

いう二つの性質を紹介する．

空円性というのは，“分割でできた三角形の外接円の内部には他の母点は含まれない”という性質である（図 1.3）．

最小角最大性というのは，“合わせると凸四角形をなす任意の二つの三角形に対して，三角形の内角の最小値が，その四角形の対角線をもう一方の対角線で置き換えてできる三角形のものと比べて小さくない” という性質である（図 1.4; 実線が Delaunay 三角形）．さらに，局所的な最小値最大性は大域的な最小値最大性と等価であることも知られている [Edelsbrunner, 1987]．すなわち，分割でできるすべての三角形の内角を小さい順に並べたリストが辞書式順序で最大になるものが Delaunay 図である．この性質のため，極端に小さい角度をもたない分割が望まれる有限要素法のためのメッシュ生成では Delaunay 図を使った分割が適しているといわれる．

1.3 位相構造と退化

母点がランダムに配置されている場合には，一つの Voronoi 点に接続している Voronoi 辺の数（Voronoi 点の次数という）は，2 次元で 3 本，3 次元で 4 本である．このことは，その双対図形である Delaunay 図においては，2 次元では Delaunay 多角形が三角形になり，3 次元では Delaunay 多面体が四面体になることに対応している．しかし，2 次元 Voronoi 図において，同一円周上に 4 個以上の母点が配置され，その円の内部に他の母点を含まない場合には，Voronoi 点の次数が 4 以上になり，Delaunay 多角形の辺は 4 本以上になる．同様に，3 次元 Voronoi 図において，同一球面上に 5 個以上の母点が配置され，その球の内部に他の母点を含まない場合には，Voronoi 点の次数が 5 以上になり，Delaunay 多面体の面数は 5 以上になる．

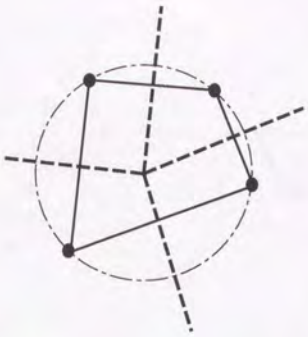


図 1.5. Voronoi 図と Delaunay 図における退化状態（2次元）

このような幾何的構造に現れる例外的状態は“退化”とよばれ，アルゴリズムを複雑にしたり，計算誤差による処理の破綻の原因になったりする．したがって，実用的なアルゴリズムを設計するためには無視できない現象である．

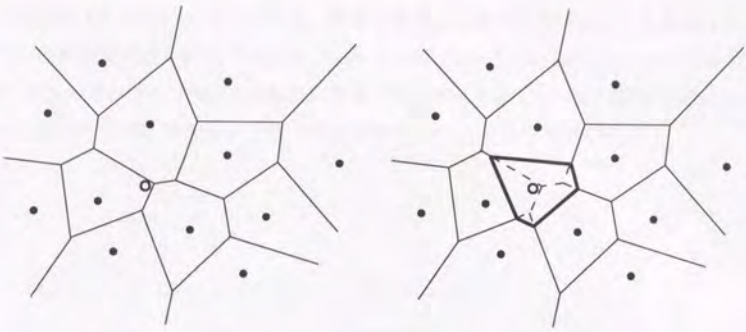
2次元の場合の退化の様子を図 1.5に示す．破線が Voronoi 辺で，実線が Delaunay 辺を表している．

1.4 逐次添加法による Voronoi 図の構成

これまで提案されている Voronoi 図構成算法のうちで実用的な意味で最も効率がよいとされているものの一つに逐次添加法 (incremental method) がある．これは，数個の母点に対する Voronoi 図から出発して，母点を一つずつ添加しながら Voronoi 図を逐次更新していく方法である．

2次元の場合の更新の様子を図 1.6 に示す．同図 (a) の Voronoi 図に対して，白丸の母点が新たに添加されたとすると，添加後の Voronoi 図は，同図 (b) のようになる．新しい母点の領域として太線の領域が新たに生成され，それに伴ない破線の部分が切りとられている．

ただし，母点逐次添加法を効率よく実行するためには，母点添加時において添加済みの母点ができるだけ空間内に一様に分布している必要がある．このために，2次元 Voronoi 図構築に対するバケットと四分木構造を使った母点の格納方法が提案されて



(a) 更新前　　(b) 更新後

図 1.6. 母点の添加時の更新の様子

おり [Ohya, *et al.*, 1984], このデータ構造を利用すれば，ランダムに配置された n 個の母点に対しては，平均的に $O(n)$ の時間（平均計算量の意味で最良である）で 平面 Voronoi 図を構成することができる．

本研究では，3 次元 Voronoi 図を対象としているため，文献 [Ohya, *et al.*, 1984] で提案された方法を自然に 3 次元へ拡張し，バケットと八分木のデータ構造を用いて母点を格納した．その後，次のような手順で母点を添加していく．

まず，母点が分布する空間を囲む直方体を同じ大きさの 8 個の小直方体に分割することを繰り返して，母点の数に比例した小直方体 — これをバケットという — に分割し，それらのバケットを葉とする八分木を用意する（図 1.7）．この八分木の各ノードに対して，それを根とする部分木の葉全体に対する領域をそのノードの支配領域と呼ぶ．各ノードにはその支配領域内の母点を 1 個ずつ格納しておく．そして，この八分木をもとに次の手順で母点の添加を行う．まず，深さ 0 の 1 個のノードの支配領域より 1 個の母点を添加する．次に深さ 1 の 8 個のノードの支配領域よりそれぞれ 1 個ずつ母点を添加する．続いて深さ 2 の 6 4 個のノードの支配領域よりそれぞれ 1 個ずつ母点を添加し，同様のことを葉の深さまで繰り返す．もしも該当する支配領域に母点がなければそこは飛ばしていく．このようにして母点添加を行なっていけば，添加済みの母点ができる限り一様に分布しているような状態を保つことができる．

ただし，逐次添加法では最悪の場合の計算量は $O(n^3)$ になる．一方，最悪の場合の

計算量を $O(n \log n)$ に抑えた算法（最悪計算量の意味で最良である）も提案されており，分割統治原理を利用する方法 [Shamos and Hoey, 1975; Lee and Schachter, 1980; Guibas and Stolfi, 1985], 凸包問題に帰着させて解く方法 [Brown , 1979; Edelsbrunner and Seidel, 1986], 平面走査を利用する方法 [Fortune, 1987] などがある．

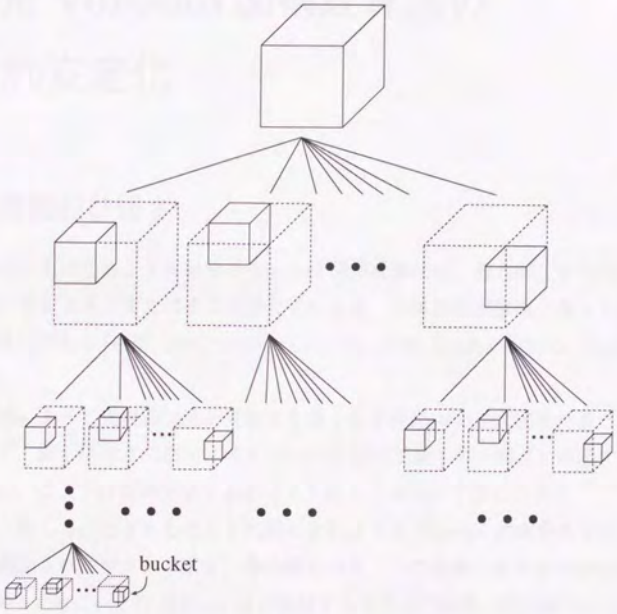


図 1.7. 8 分木

第 2 章

3 次元 Voronoi 図構成算法の数値的安定化

2.1 数値的ひ弱さ

逐次添加法をはじめとする従来の Voronoi 図構成算法は，母点同士が退化を生じる位置関係にあるとき，またはその状態に近いとき，計算誤差が原因となって処理が破綻する危険性がある [杉原, 1991; Sugihara and Iri, 1992; Sugihara, 1992; Sugihara and Iri, 1994].

計算誤差によって逐次添加法が破綻する様子を 2 次元 Voronoi 図を対象として説明する．まず，新しい母点の添加に伴う Voronoi 図の更新の例を図 2.1 に示す．前章でも示したが，ここでは計算方法を含めてより詳しくみていくことにする．

最初に，新しく添加された母点を内部に含むような Voronoi 領域を見つけ，対応する母点と添加された母点との垂直二等分線を引き，この領域に属する Voronoi 辺との交点を求める．次に，その Voronoi 辺に隣接する母点との垂直二等分線を引く．この操作を繰り返すことで，閉じた多角形領域が得られる．最後にその内部の構造を取り去ることにより，新しく添加された母点の Voronoi 領域が得られる．このことは Voronoi 図の定義より容易にわかる．

ただし，この処理には例外がある．新しく添加する母点が，それまでに添加された母点集合の凸包上または外側にある場合には，図 2.2 に示すように，新しい母点の領域は閉じた領域にならないので，その場合には上記の操作を少し変更しなければならない．しかし，次の便法を利用することで，このような例外処理を不要にすることが

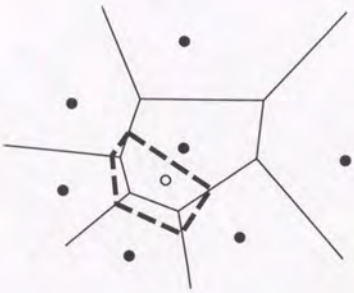


図 2.1. 母点の添加による Voronoi 図の更新

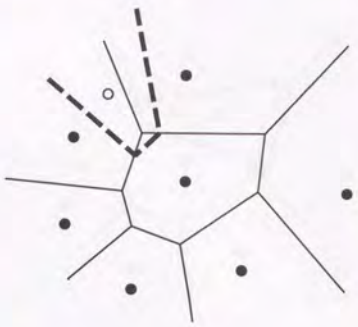


図 2.2. 新しい母点の Voronoi 領域の境界が閉じない例

できる．

図2.3に示すように，与えられた母点を全て含む十分に大きな三角形を考え，各頂点に母点を配置する．こうして，便宜上加えた3個の母点に対する Voronoi 図から出発して，逐次添加法を行なっていけば，図 2.2 のような例外が生じることはない．以下ではこの便法を採用するものとして話を進める．

さて，上で述べたように，逐次添加法における Voronoi 図更新作業においては，新しい Voronoi 辺と既存の Voronoi 辺との交点を求めるために数値計算が行なわれる．しかし，通常は計算誤差のためにこれらの点の位置は近似的に求まるだけである．よって，たとえば図 2.4 に示すように，添加された母点の領域の境界が閉じないという結果が起こり得る．このまま処理を進めれば，Voronoi 領域は閉じているという性質に根拠をおいた処理は破綻してしまう．

3次元の場合にも同様の議論ができる．まず例外処理をなくすために，図 2.5に示すような与えられた母点を全て含む十分大きな四面体を考え，その 4 個の頂点に便宜上母点を配置する．そして，この 4 個の母点に対する Voronoi 図から出発して逐次添加法を行なう．このような便法を採用すると，添加される母点の領域は必ず閉じた領域になる．しかし，計算誤差を含んだ近似的な数値計算を行なうと，2次元の場合と同様に領域の境界が閉じない場合があり得る（この状況をわかりやすく図示するのは難しいが，2次元の場合からの類推は容易にできるであろう）．こうなると Voronoi 図の位相構造に矛盾が生じ，処理が破綻してしまう．

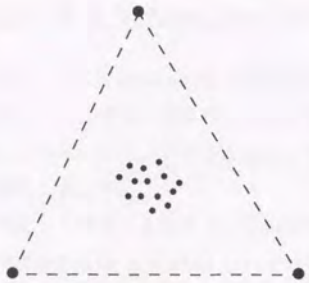


図　2.3.　便宜上加えた母点（２次元 Voronoi 図の場合）

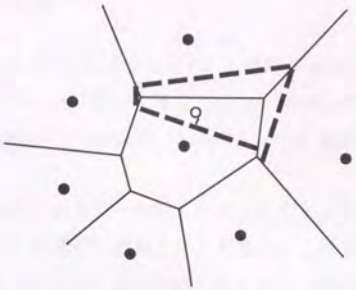


図 2.4. 数値誤差によって破綻する例

したがって，このようなアルゴリズムをそのまま計算機プログラムに翻訳しても数値的に不安定であり，実用的なシステムを構築することはできない．

本章では，逐次添加法を基に，「位相優先法」の考え方を採り入れて数値的安定化を計った３次元 Voronoi 図の構成アルゴリズムを提案する [Inagaki, Sugihara and Sugie, 1992; 稲垣, 杉原, 杉江, 1994]．３次元 Voronoi 図においては，２次元 Voronoi 領域（多角形）の境界と異なり，Voronoi 領域（多面体）の境界に自然な順序がつかないなどの差があるため，「位相優先法」を適用する際に必要な位相的性質の保持などは，２次元の場合を単純に延長することができず，本質的に新しい工夫が必要となってくる．その詳細を次節以降で説明する．なお，２次元 Voronoi 図を対象とした数値的に安定な構成法については，文献 [杉原, 1991; Sugihara and Iri, 1994] を参照されたい．

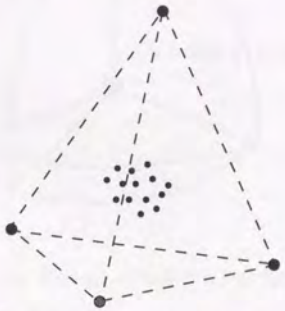


図 2.5. 便宜上加えた母点（３次元 Voronoi 図の場合）

2.2 3次元 Voronoi 図の位相構造

まず，3次元 Voronoi 図の位相構造を見てみる．図2.6に示すように，1個の Voronoi 点に対し，通常は，4本の Voronoi 辺，6個の Voronoi 面，4個の Voronoi 領域が隣接している．また，1本の Voronoi 辺には，3個の Voronoi 面，3個の Voronoi 領域が隣接している．

退化した場合（5個以上の母点が同一の球面上にあり，その球の内部に他の母点がない場合）には，Voronoi 点のまわりの Voronoi 領域が5個以上になるなど，上の通常の場合からはずれる．しかし，位相優先法においては，数値誤差のある世界で退化しているか否かを厳密に判定することはできないという立場でアルゴリズムを設計するので，退化のない場合だけを扱えば十分である．

Voronoi 図は次のような位相的性質を持つ．

P1 それぞれの母点は空でない Voronoi 領域を持つ．

P2 Voronoi 領域は単連結である．

P3 二つの Voronoi 領域の境界は，高々1個の Voronoi 面を共有する．

P1 は定義から明らかであり，P2，P3 は Voronoi 領域が凸であることから導かれる．

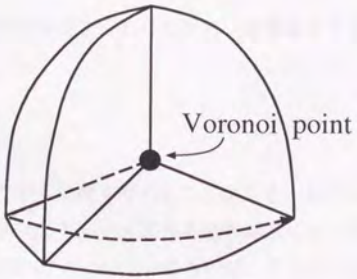


図 2.6. Voronoi 点のまわりの位相構造

ただし，これらの位相的性質は，3次元 Voronoi 図であるための必要条件でしかない．今のところ，3次元 Voronoi 図を位相的性質だけで特徴づける必要十分条件はわかっていない（2次元 Voronoi 図においても同様である）．そこで，3次元 Voronoi 図が満たすべき位相的性質のうち，純粋に組合せ的でかつそれが満たされているかどうかを効率良く判定できるものをできるだけ多く選んだ結果が，上記の P1 ～ P3 なのである．

2.3 数値的安定化のための基本方針

3次元 Voronoi 図の構成過程において処理が破綻するのは，計算誤差のために判定を誤り，3次元 Voronoi 図が有するべき位相構造が破壊される場合である．幾何図形処理においては，常にその図形の持つ位相的性質に矛盾が生じないように処理を進める必要がある．そこで，位相優先法を適用し，3次元 Voronoi 図構成手続きの主要部分を位相構造の更新と捉え，その更新が矛盾なく行なわれることを最優先する．もし，Voronoi 図が満たすべき位相的性質に反するような数値判定がなされた場合には，それは採用しないことにする．そして，位相的に矛盾の起きない範囲でなお残る任意性を解消するためにのみ，数値計算結果を用いる．

2.4 データ構造とアルゴリズム

前節で述べた基本方針を実現するための，データ構造とアルゴリズムについて説明する．

2.4.1 データ構造

位相的性質を確認しながら処理を進めることができ，母点添加時の更新作業も効率良く行なえるように，3次元 Voronoi 図の各図形要素に持たせるデータを次のように決めた．データ項目の右肩に*がついているものは，3次元 Voronoi 図の計量的性質に関するデータであり，その他のものはすべて位相構造に関するデータである．また，各図形要素近傍の位相構造を図 2.7～2.10 に示した．

Voronoi 点

- その Voronoi 点の x, y, z座標値*
- その Voronoi 点に接続する 4 個の Voronoi 辺

Voronoi 辺

- その Voronoi 辺の端点となる 2 個の Voronoi 点
  （辺は有効線分で表わされ，始点・終点の区別あり）
- その Voronoi 辺に接続する 3 個の Voronoi 面

Voronoi 面

- その Voronoi 面に属する Voronoi 辺のうちの 1 個
- その Voronoi 面の両側に接続する Voronoi 領域

Voronoi 領域

- その Voronoi 領域を勢力圏とする母点の x, y, z座標値*
- その Voronoi 領域に属する Voronoi 面のうちの 1 個

各図形要素に上記のような位相データを持たせることで，3 次元 Voronoi 図内を自由に探索することが可能となる．例えば，一つの Voronoi 領域（多面体）の境界となっ

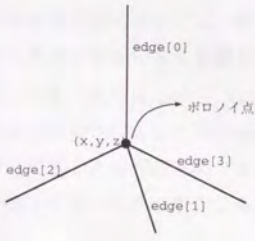


図 2.7. Voronoi 点近傍の様子

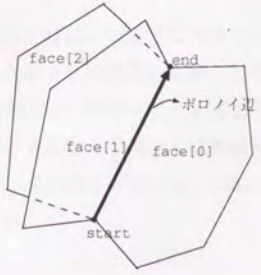


図 2.8. Voronoi 辺近傍の様子

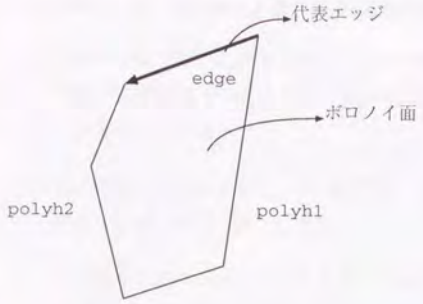


図 2.9. Voronoi 面近傍の様子

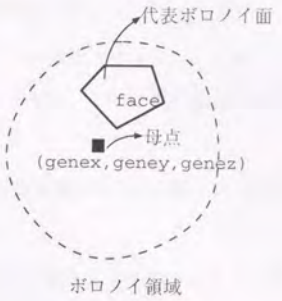


図 2.10. Voronoi 領域近傍の様子

ている Voronoi 面の集合のように，自然な順序づけができない図形要素を検索する場合にも，各図形要素に持たせてある位相データに基づき，次のような操作を行なう．

まず，その Voronoi 領域に属する一つの Voronoi 面をキューに入れる．次に，キューから Voronoi 面を一つ取り出し，その Voronoi 面に隣接する面のうちで，対象としている Voronoi 領域の境界となっており，かつ未検索のものをキューに入れる．これをキューが空になるまで繰り返す．

以上の操作により，一つの Voronoi 領域（多面体）の境界となっているすべての Voronoi 面が，キューから順次取り出されることになる．このようにすることで，一つの Voronoi 面から始めて，そのまわりに徐々に検索範囲を拡大していき，最終的には，多面体全体を包み込むようにして検索が行なわれる．

2.4.2 位相構造に着目した Voronoi 図更新手続き

母点逐次添加法においては，新しい母点が添加された時にそれまでにできている図形をどう更新するかが最も重要な作業である．この作業を位相構造の更新として捉えてみる．いま，母点 $p_1, p_2, \cdots, p_{l-1}$ に対する Voronoi 図 Vor^{l-1} が得られているとし，新たな母点 p_l の添加によって Voronoi 図 Vor^{l-1} を Vor^l に更新する手続きを位相構造のみに着目して捉えたのが以下のアルゴリズム A である．その様子を図 2.11 に順に示す．四角で表されているのが新たに添加された母点である．

アルゴリズム A [**Voronoi 図の位相構造の更新手続き**]

1. Voronoi 図 Vor^{l-1}に属するある Voronoi 点集合 Tを選ぶ（図 2.11(a); ただしここでは，図が複雑でわかりにくくなるのを避けるために，Tの要素が 1 点の場合について示している）.

2. Tに属する Voronoi 点と，Tに属さない Voronoi 点を結ぶ辺上に，新しい Voronoi 点を 1 つずつ作る (図 2.11(b)).

3. 2 で作られた Voronoi 点のうち同じ Voronoi 領域に属する点を順に結んで，新しい Voronoi 面を作る (図 2.11(c)).

4. Tに属する Voronoi 点と，それらを結ぶ辺，及びそれらで張られる Voronoi 面で作られる Vor^{l-1}の部分構造を削除する (図 2.11(d)).

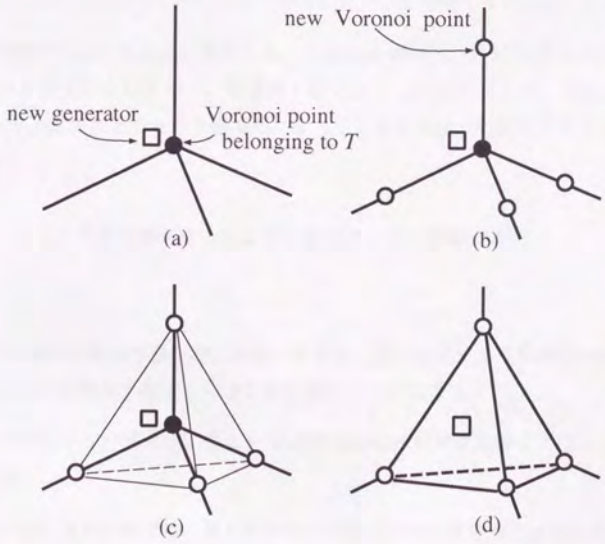


図 2.11. 母点添加時の位相的変化

ただし，3次元 Voronoi 図の位相的性質として選んだ **P1** ～ **P3**（2.2節参照）が満たされるように，削除される Voronoi 点集合 T を選ばなければならない．そのために T が満たすべき位相的性質として，以下の **C1** ～ **C5** をチェックすることにする．

削除すべき Voronoi 点集合の位相的性質

C1: T は空集合でない．

C2: Tに属する点とそれらを結ぶ辺によってできる Vor^{l-1}の部分グラフは単連結である．

C3: ひとつの Voronoi 領域に属するすべての Voronoi 点が，Tの要素であることはない．

C4: Tの要素のうち Voronoi 領域 $V(p_i)$ に属するものの集合を $T(p_i)$ と表すことにすると，任意の $i=1,2,\cdots,l-1$ に対して，$T(p_i)$ の要素である Voronoi 点とそれらを結ぶ辺によってできる Vor^{l-1}の部分グラフも連結している．

C5: Tの要素でない Voronoi 点のうち，Voronoi 領域 $V(p_i)$ に属するものの集合を $\overline{T}(p_i)$ と表すことにすると，任意の $i=1,2,\cdots,l-1$ に対して，$\overline{T}(p_i)$ の要素である Voronoi 点とそれらを結ぶ辺によってできる Vor^{l-1}の部分グラフも連結している．

以下に，これらの条件の必要性および十分性について考察する．

必要性

Tがこの5個の位相的性質を満たさないときは，以下に示すような事態が生じ，3次元 Voronoi 図の位相的性質 **P1** ～ **P3** と矛盾することになる．

1. **C1** を満たさない場合は，新しい母点の Voronoi 領域が空集合となって，**P1** に反する．
2. **C2** を満たさない場合は，新しい母点の添加によってできる Voronoi 領域が2個以上の多面体に分裂し，**P2** に反する（図 2.12(a); 網のかかった領域は，削除される Voronoi 点集合が存在している領域を表しており，同図 (b)(c)(d) も同様である）．

3. **C3** を満たさない場合は，既存の Voronoi 領域が，新しい母点の添加によって完全に消滅することになり，**P1** に反する（図 2.12(b)）．

4. **C4** を満たさない場合は，既存の Voronoi 領域において，新しい母点の添加によって切り取られる部分がその領域内で 2 カ所以上に分かれていることになり，**P3** に反する（図 2.12(c)）．

5. **C5** を満たさない場合は，既存の Voronoi 領域が新しい母点の添加によって 2 個以上に分断され，**P2** に反する（図 2.12(d)）．

したがって，必要条件になっていることがわかる．

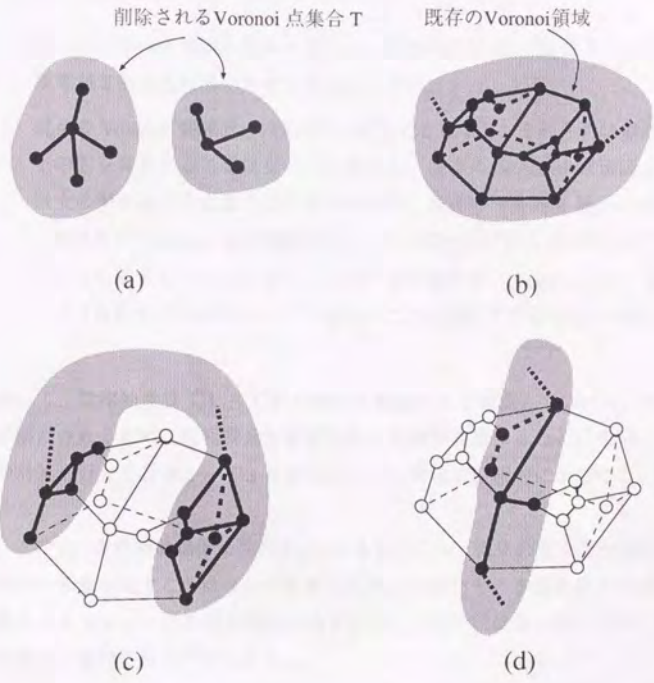


図 2.12. 位相構造の破壊を引き起こすような Tの選び方

十分性

1. **P1** が成り立つことが以下のことよりわかる．

   (a) 新しい Voronoi 領域は，**C1** により，空にならない．

   (b) 既存の Voronoi 領域は，**C3** により，空にならない．

2. **P2** が成り立つことが以下のことよりわかる．

   (a) 既存の Voronoi 領域は，**C5** により，単連結でなくなることはない．

   (b) 新しい Voronoi 領域は，**C2, C4** により，単連結である．

3. **P3** が成り立つことが以下のことよりわかる．

   (a) 新しい Voronoi 領域と既存の Voronoi 領域の間では，**C4** により，それらの境界が 2 個以上の面を共有することはない．

   (b) 既存の Voronoi 領域同士の間では，**C4, C5** により，それらの境界が 2 個以上の面を共有することはない．なぜなら，既存の Voronoi 領域同士が 2 個以上の面を共有するようになるためには，既存の Voronoi 面上において T に属さない Voronoi 点が非連結になっていなければならないが（図 2.13; 黒丸が T に属する Voronoi 点で，白丸が T に属さない Voronoi 点），**C4, C5** によりそのような状況にはなり得ないことが保証されるからである．

したがって，位相的性質 **C1** ～ **C5** を満たす範囲内で T を選んでいけば，常に **P1** ～ **P3** が満たされるため，位相構造が破壊されて処理が破綻することはない．いかに低精度の計算を行っても **P1** ～ **P3** を満たすという意味で位相的に矛盾のない結果が出力される．

また，このような位相的無矛盾性のチェックを加えても，従来の逐次添加型算法と比べて平均的計算量が増すことはないと予想される．なぜなら，1 個の母点を添加したとき変更される Voronoi 図の部分構造の大きさは，平均的には母点数に依存しない一定規模であると期待できるからである．

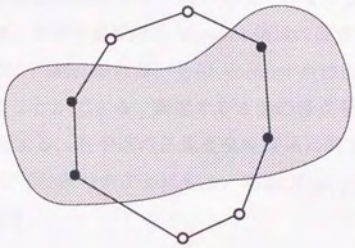


図 2.13. 既存の Voronoi 面が分裂するような T の選び方

2.4.3 数値判定の利用

性質 C1 ～ C5 を満たす Voronoi 点集合 Tは多数ある．そのうち，Voronoi 図の更新手続きのために採用すべきものがどれであるかを決定しなければならない．Tに属するべき Voronoi 点は，新しい母点の Voronoi 領域に含まれるものである．したがって，その判定は（退化も数値誤差もなければ）次の方法で行なうことができる．

判定したい Voronoi 点に隣接する 4 個の Voronoi 領域の母点を通る球を考え，新しく添加された母点がその球の内部にあれば，その Voronoi 点は削除され，外部にあれば，削除されないと判定することができる．隣接する 4 個の母点を p_i, p_j, p_k, p_lとし，新しく添加された母点を p とする．右手系の正規直交座標系に関する p の座標を (x, y, z)，p_iの座標を (x_i, y_i, z_i)，などと表わすことにする．実数 $H(p_i, p_j, p_k, p_l, p)$ を

$$H(p_i, p_j, p_k, p_l, p)$$

$$= \begin{vmatrix} 1 & x_i & y_i & z_i & (x_i^2+y_i^2+z_i^2)/2 \\ 1 & x_j & y_j & z_j & (x_j^2+y_j^2+z_j^2)/2 \\ 1 & x_k & y_k & z_k & (x_k^2+y_k^2+z_k^2)/2 \\ 1 & x_l & y_l & z_l & (x_l^2+y_l^2+z_l^2)/2 \\ 1 & x & y & z & (x^2+y^2+z^2)/2 \end{vmatrix} \cdot \begin{vmatrix} 1 & x_i & y_i & z_i \\ 1 & x_j & y_j & z_j \\ 1 & x_k & y_k & z_k \\ 1 & x_l & y_l & z_l \end{vmatrix} \tag{2.1}$$

と定義する．4 点 p_i, p_j, p_k, p_lが同一平面上にないとき，$H(p_i, p_j, p_k, p_l, p) = 0$ を満たす p は 4 点を通る球面上にあり，

$H < 0$ ならば球の内部，

$H > 0$ ならば球の外部にある

と判定する．

したがって，“原理的” には Hの符号に基づいて，各 Voronoi 点が Tに属するか否かを判定できる．しかし，実際には数値誤差があるため，Hの符号に頼りきることはできない．そこで，C1 ～ C5 という位相的性質が満たされることを最優先し，それに反しない範囲で一番もっともらしい Tを選ぶための優先度の低い情報として Hの符号を利用する，というのが本稿で提案する数値的安定化の骨子である．

p_i, p_j, p_k, p_lの Voronoi 領域の境界が共有する Voronoi 点を vとし，新しく添加する母点を p とするとき，$H(p_i, p_j, p_k, p_l, p)$ を，以下では $H(v)$ と略記することにする．

2.4.4 位相的性質をチェックするための手続き

2.4.2節で述べたアルゴリズム A を利用する場合に最も難しい部分が，位相的性質 C1 ～ C5 を満たすような削除すべき Voronoi 点集合 T を選び出すところであろう．ここでは，その部分のアルゴリズムについて説明する．

アルゴリズムの記述を簡潔にするために一つの準備をする．1 個の Voronoi 領域の境界上の Voronoi 点と Voronoi 辺は，前者を頂点とし後者を枝とするグラフ G とみなすことができる．このグラフ G の頂点の集合を N とする．$v_1, v_2 \in N$，$N' \subseteq N - \{v_1, v_2\}$ に対して，グラフ G において v_1 と v_2 をつなぐ全ての道が，N' の要素を少なくとも 1 個含むとき，v_1 と v_2 は N' によって隔てられているということにする．

次に述べるアルゴリズムでは，残すことに決めた Voronoi 点集合を T_O，T に隣接する頂点のうち T にも T_O にも属さない Voronoi 点集合を T_N で表すことにする．T_N を表現するにはキューを用いる．したがって，T_N から要素を取り出すときには，最も古いものから順に取り出す．はじめは，T，T_O，T_N はすべて空集合である．

アルゴリズム B [削除すべき Voronoi 点集合 T を選ぶアルゴリズム]

1. 新しく添加された母点に最も近い母点の Voronoi 領域が成す多面体を見つける．
2. その多面体に属する頂点のうち，H の値が最小となるものを，T の最初の要素とし，その頂点に隣接する全ての頂点を T_N に入れる．
3. T_N から最も古い要素 v を取り出す．
4. v に関して，以下の 5 ～ 7 のステップが既に 1 回以上実行されており，最後に実行されたときの集合 T と T_O が，現在のものと同じであったなら，v を T_O に入れて，8 へ行く．
5. $H(v)$ を計算する．
6. $H(v) \geq 0$ の場合：すぐに "残留" と決定して，v を T_O に入れ，8 へ行く．
7. $H(v) < 0$ の場合：
   (a) v が載っている 4 個の多面体に関して，1 つでも以下の i) または ii) または iii) の条件に当てはまれば，v を T_O に入れる（すなわち，数値判定結果を覆し，"残留" と決定する）．

i) v 以外の頂点が全て T に属す

ii) Tに属す頂点と vとが，T_O によって隔てられている．

iii) v を T に入れると，T_O に属す2個の頂点で，Tによって隔てられてしまうものがある．

(b) 上記 i),ii),iii) の条件には当てはまらないが，v が載っている4つの多面体に関して，1つでも以下の iv) または v) の条件に当てはまれば，vを T_Nに戻す（決定を "保留" する）．

iv) Tに属す頂点と vとが，$T_O \cup T_N$ によって隔てられている．

v) v を T に入れると，$T_O \cup T_N$ に属す2個の頂点のうちTによって隔てられてしまうものがある．

(c) 上記 i),ii),iii),iv),v) のどれにも当てはまらない場合にのみ，vを T に入れ（"削除" と決定する），vに隣接し$T \cup T_O \cup T_N$に属さない頂点を全て T_Nに入れる．

8. T_Nに属するものがなくなるまで，3 ～ 7 を繰り返す．

このアルゴリズムでは，ステップ2で位相的性質 C1 を保証し，それ以降は，Tに属す頂点に隣接するものしか見ていないので，位相的性質 C2 のうち連結性を満たす．また，ステップ7の i) で C3 をチェックし，ii),iv) で C4 を，iii),v) で C5 のチェックを行なっている．

ここで，iv) および v) においては，未決定の頂点も併せた位相的チェックを行なっており，後の判定次第では位相的性質を満たさなくなる恐れがある場合には，決定を保留し，確定できるところから決めていくことにしている．iv) における処理は，C2 の単連結性をも保証している．穴のあいた形状ができる可能性があるときには，決定を保留して，穴になる可能性のあるところを先に決定しようとしていることになる．これらの処理は，2次元 Voronoi 図の構成アルゴリズムを3次元へ拡張するにあたり，新しく必要になった処理であるので，以下に説明を加える．

2次元 Voronoi 図の構成過程においては，ひとつの Voronoi 領域（多角形）の境界上の Voronoi 点のうちで T に属するものと，それらを結ぶ Voronoi 辺で作られる部分グラフが，非連結になるような数値判定がなされた時には（図 2.14(a)），即座に数値判定結果を覆す．なぜなら，2次元の場合には，Tに属する Voronoi 点とそれらを

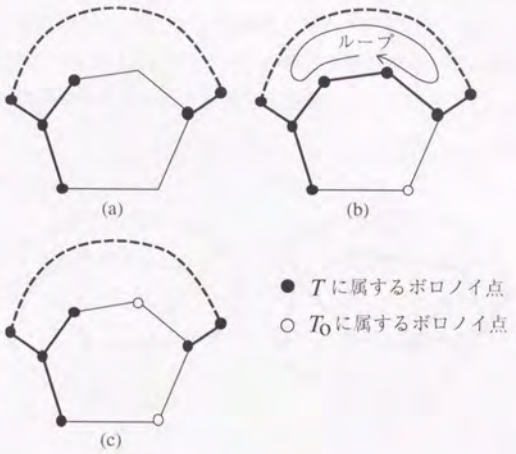


図 2.14. 2次元 Voronoi 図の構成過程における Tの位相的性質のチェック

結ぶ Voronoi 辺で作られる部分グラフには閉路があってはならないため [杉原, 1991; Sugihara and Iri, 1994]，図 2.14(b) のようになることはない．よって，それ以降の判定がどのようになされようとも，その Voronoi 領域上の部分グラフは非連結のままであり（図 2.14(c)），Tが有するべき位相的性質を満たさなくなるからである．

一方，3次元の場合には，新しい母点の添加により，既存の Voronoi 面が消滅してしまうことがあるため，Tに属する Voronoi 点とそれらを結ぶ Voronoi 辺で作られる部分グラフには，閉路が含まれていてもよい．このため，ひとつの Voronoi 領域（多面体）の境界上の Voronoi 点のうちで T に属するものと，それらを結ぶ Voronoi 辺で作られる部分グラフが，非連結になるような数値判定がなされた時でも（図 2.15(a)），その後の判定結果次第では，図 2.15(b) のように，Voronoi 領域上の部分グラフが連結となる場合もあれば，図 2.15(c) のように，非連結のままである場合もある．したがって，その時点で決定してしまうと，あとでその決定を覆さなければならなくなることがある．そこで，本アルゴリズムにおいては，決定を一時 " 保留" して，他の Voronoi 点の決定を待った後に，決定することにしている．この工夫により，位相的性質 C1 ～ C5 を満たす Tをいつでも効率よく得ることができる．

なお，このアルゴリズムでは退化に対する例外処理が全く含まれていないことにも注意されたい．すなわち，$H(v) \geq 0$ の場合（ステップ 6）と，$H(v) < 0$ の場合（ステップ 7）に分けているだけで，$H(v) = 0$ の場合を特別扱いしていない．したがって，アルゴリズムの構造は従来のものより著しく簡単になっている．

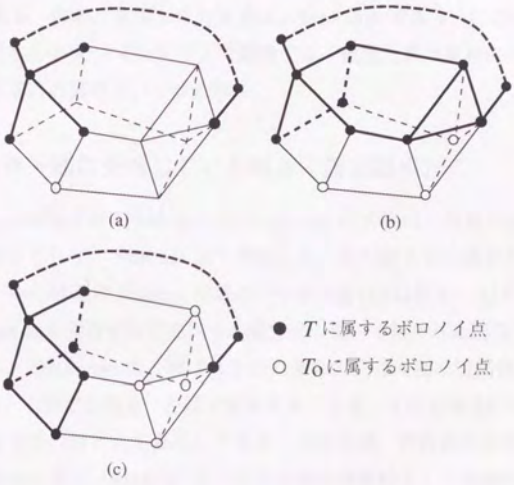


図 2.15. 3次元 Voronoi 図の構成過程における Tの位相的性質のチェック

2.5 計算機実験

2.5.1 アルゴリズムの実装

C言語を用いて本アルゴリズムの計算機への実装を行なった．プログラムの規模は，C言語のソースプログラムで5000行程度であり，そのうち，入出力処理・八分木構造への母点格納処理などを除いた，母点添加時のVoronoi図更新手続きに関する部分は，2500行程度である．実験に使用した計算機は，Sun4/370であり，UNIXオペレーティングシステムのもとのXウインドウ上で稼働する．数値計算は単精度浮動小数点表現で行なった．以下に計算例をいくつか示す．

2.5.2 母点が一様に分布している場合（母点数小）

3次元空間内の単位立方体領域 $[0,1) \times [0,1) \times [0,1)$ の内部に乱数を用いて一様に発生させた点を母点として，Voronoi図を構成した．母点数50の場合の出力結果を図2.16(a)に示す．中心付近のVoronoi領域の一つを同図(b)に示す．(a)のVoronoi図において，Voronoi面を共有する母点同士を線分でつないで得られる構造を同図(c)に示す．この構造は，Delaunay図と呼ばれる母点集合の凸包内部の四面体分割を与える．

これらの図は，右眼で左側を，左眼で右側を見たとき，3次元構造が知覚できる（両眼立体視）ような図の対として表示してある．これ以降，計算機の出力として得られた3次元のVoronoi図／Delaunay図の表示方法には原則として同様の方法を採用することにする．

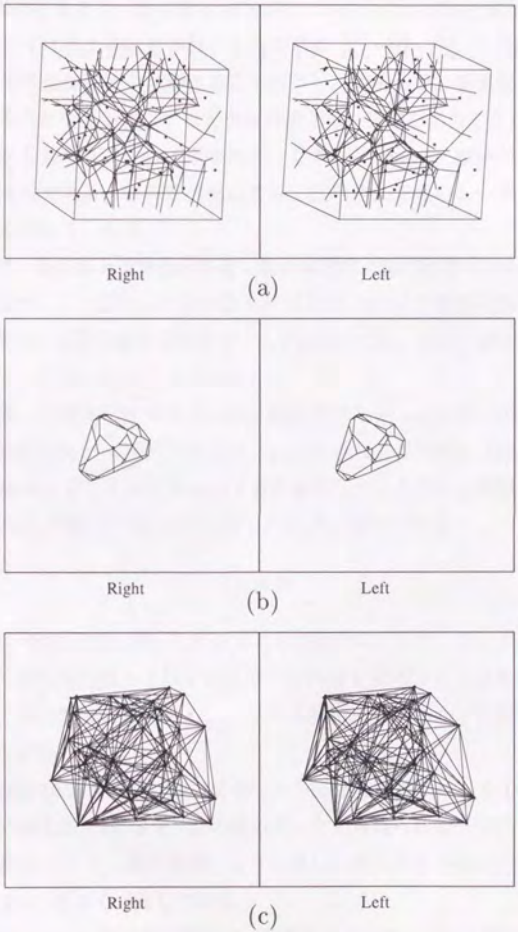


図 2.16. 母点が一様に分布している場合 (母点数 50): (a) Voronoi 図; (b) 中心付近の一つの Voronoi 領域; (c) Delaunay 図

2.5.3 母点が一様に分布している場合（母点数大）

前節と同じ条件のもとで，母点数を1000個まで増やした．出力された3次元 Voronoi 図を図2.17に示す（ここでは立体視による表示はしていない）．この結果を見ても3次元 Voronoi 図の全体構造を把握することは不可能であるが，重要なことは，途中で処理が破綻することなく正常にプログラムが終了し，結果を出力していることである．この出力の中からからランダムに選ばれたいくつかの Voronoi 領域のみを表示させてみたものが図2.18である．母点を内部に含み，位相的に矛盾のない多面体領域が生成されていることがみてとれる．

この実験では，母点逐次添加法の処理に要する時間を途中経過も含めて測定した．結果を図2.19に示す．この図から，母点数 n に対して，$O(n)$ の処理時間で実現されていることが確認でき，位相的無矛盾性のチェックを入れても，従来の算法に比べ著しく計算効率が悪くなってはいないことがわかる．

また，母点数 n の増加に伴なう Voronoi 図全体の Voronoi 点数・ Voronoi 辺数を測定した．測定結果を表2.1に示す．3次元 Voronoi 図は，前述のように退化のない場合には一つの Voronoi 点に4本の Voronoi 辺が隣接しているという位相構造を有しているので， Voronoi 点数 v と Voronoi 辺数 e には次の関係がある．

$$e = 2v$$

表2.1を見ると，常に上式を満たしていることがわかる．また，新しく添加した母点の Voronoi 領域の平均の面数および1面当たりの角数を求めてみた結果が表2.2である．これをみると，母点数 n が増えても，一つの Voronoi 領域あたりの図形要素数はほぼ一定であることがわかる．

さらに，母点数1000の場合について全ての Voronoi 領域に対して1多面体当りの面数を調査した結果を表2.3に示す．この表のデータから得られる一つの Voronoi 領域当たりの平均面数は14.8で，逐次添加によって新しく作られた Voronoi 領域の面数の平均値14.9（表2.2）とよく一致している．

なお，3次元 Voronoi 図の面の総数は，最悪の場合には，$O(n^2)$ 個となる（したがって，1多面体当たり $O(n)$ 個）ことが知られているが，ランダムに分布する母点に対して $n \leq 1000$ の範囲では，そのような傾向は全く見られないことが，この表（表2.3）からわかる．

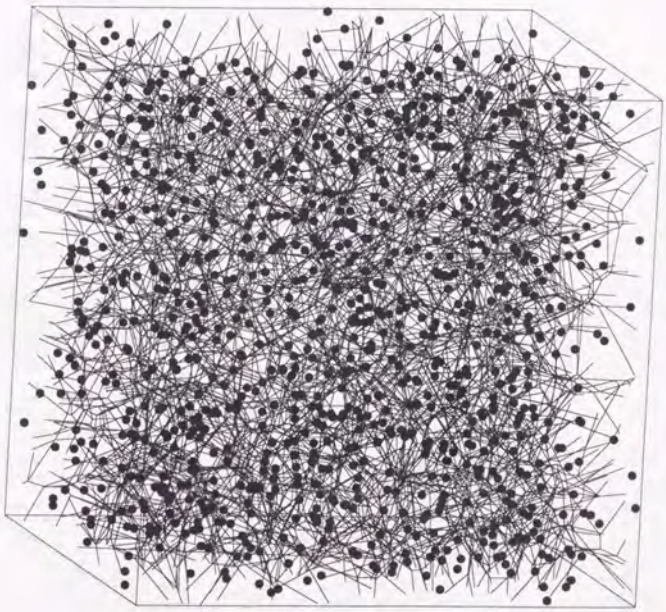


図 2.17. 一様分布の 1000 個の母点に対する Voronoi 図

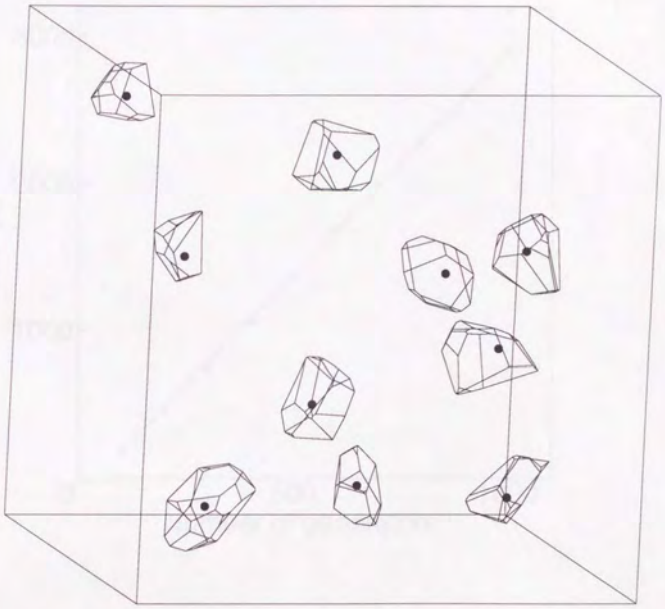


図 2.18. ランダムに抽出した Voronoi 領域

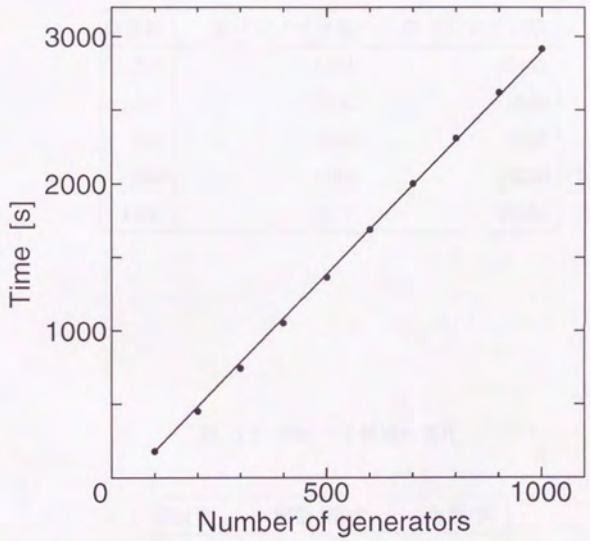


図 2.19. 計算時間の測定

表 2.1. 図形要素数の変化

母点数	総ボロノイ点数	総ボロノイ辺数
200	1221	2442
400	2520	5040
600	3834	7668
800	5165	10330
1000	6477	12954

表 2.2. ボロノイ領域の変化

母点数	面数/領域	角数/面
200	13.9	5.1
400	14.4	5.2
600	14.7	5.2
800	14.8	5.2
1000	14.9	5.2

表 2.3. ボロノイ領域のサイズ

面数／領域	該当する領域数
6	1
7	5
8	10
9	26
10	51
11	66
12	104
13	122
14	125
15	113
16	87
17	86
18	66
19	43
20	32
21	27
22	13
23	9
24	5
25	8
26	0
27	1

2.5.4 母点が同一球面上に分布している場合

著しく退化している状態を作るため，$[0,1)\times[0,1)\times[0,1)$ の立方体領域に内接する球面上に乱数を用いて一様に発生させた点を母点とした．この場合においても処理が破綻することなく最後まで実行された．従来の数値判定のみに基づいた算法では，このような入力に対して，大抵の場合処理が破綻する．母点数を 100 としたときの出力結果を図 2.20(a) に示す．中心付近に Voronoi 点が集まり，著しく退化していることがわかる．この中心付近を非常に拡大してみたのが，同図 (b)(c) である．倍率は (b) が 1×10^6，(c) が 3×10^7である．これを見ると，中心付近は真の Voronoi 図とはかなりかけ離れていることがわかる．しかし，位相的には，P1 ～ P3 を満たすという意味で矛盾のないことが理論的に保証されている．この場合のように，間違いのない数値判定をすることがほぼ不可能な状況においても，処理が最後まで実行されることが確認でき，本算法の数値的安定性が実証された．

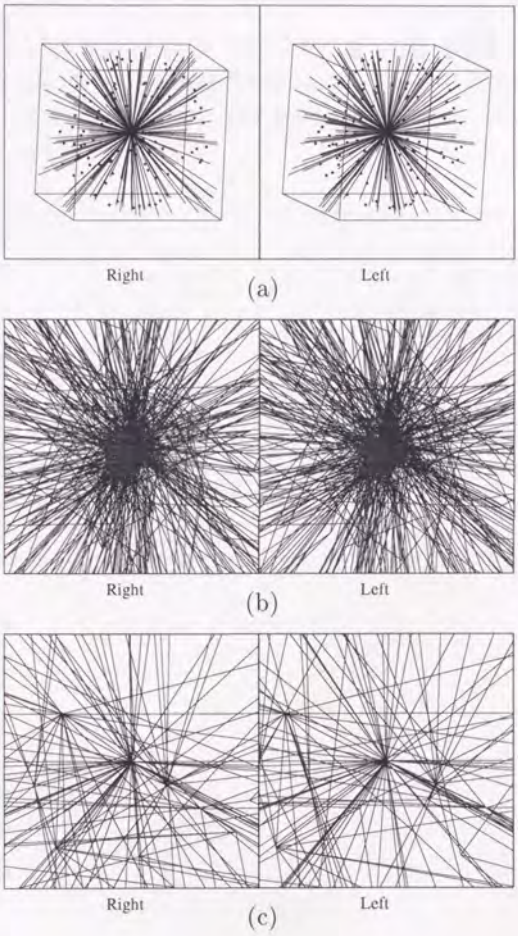


図 2.20. 母点が同一球面上に分布している場合(母点数 100): (a) Voronoi 図; (b) 中心付近の拡大図 (1×10^6); (c) 中心付近の拡大図 (3×10^7)

2.5.5 数値判定を乱数に置き換えた場合

計算誤差の究極の場合として，プログラムの中で判定のために用いている $H(v)$ の値を乱数に置き換えて実行してみた．この場合においても処理は最後まで実行され，結果を出力した．母点数 10 の場合の出力結果を図 2.21に示す．結果はもちろん真の Voronoi 図とはかけ離れてはいるが，位相的には矛盾のない処理を行なっているので，途中で破綻することはない．

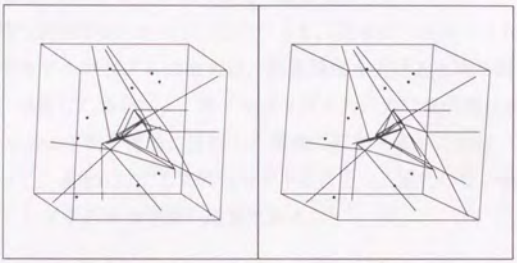


図 2.21. 数値判定 H を乱数に置き換えた場合 (母点数 10)

2.6 考察

母点逐次添加法による3次元 Voronoi 図構成算法を改良し，数値的安定化を計った．ここでの基本方針は，母点添加時の Voronoi 図更新作業の位相的側面に着目し，数値計算結果より位相構造の無矛盾性を優先させることによって，計算誤差による処理の破綻を防いだものである．

退化のない入力データに対して無限の精度で数値計算を行なった場合には，本アルゴリズムの位相的チェックは実質的には何の働きもしない（すなわち，数値計算のみに頼っても同じ判定結果が得られる）．したがって，従来のアルゴリズムと同じ結果を出す．この意味で本アルゴリズムの出力は，計算精度を上げると真の解に収束すると言える．ただし，退化した場合には，本アルゴリズムの出力は真の解に長さ0の Voronoi 辺や面積0の Voronoi 面を適当に追加した構造に収束する．これは，位相構造を優先することによって，退化に対する例外処理が不要になり，退化のない構造のみを扱っているという本アルゴリズムの特徴の結果である．

第 3 章

3 次元 Delaunay 図の構成

第 2 章において，3 次元 Voronoi 図構成のための逐次算法を位相優先法に基づいて設計しなおすことにより，数値的に非常に安定した計算機プログラムが実現されることを示した．本章では 3 次元 Voronoi 図の双対図形である 3 次元 Delaunay 図の生成について述べたい．

理論的には一旦 Voronoi 図が得られれば，その位相構造にしたがって双対を求めていけば Delaunay 図が得られるはずである．そこで，数値的安定化が計られた 3 次元 Voronoi 図構成プログラムを利用して 3 次元 Delaunay 図の生成実験を繰り返し行なった．すると，退化した母点配置を与えた場合に，互いに重なりあった Delaunay 四面体が出力されていることがわかった．

その原因を追求してみたところ，それは単なるプログラムの設計ミスではなく，3 次元 Voronoi 図の構成では表面化しなかった 3 次元 Delaunay 図特有の難しさがあることがわかってきた [稲垣, 杉原, 1994a: Sugihara and Inagaki, 1995]．計算誤差のある世界では，3 次元の Voronoi 図と Delaunay 図はその構成における難しさには差があるのである．

3.1 3 次元 Delaunay 図の生成

母点がランダムに配置されている状態であれば，Voronoi 図，Delaunay 図の両図形ともその頂点数，辺数，面数，領域数は $O(n)$ （n は母点数）であり，したがって，互いの変換に要する手間は $O(n)$ である．

3 次元 Voronoi 図の出力例を図 3.1に示す．この例では，3 次元空間内に乱数を使っ

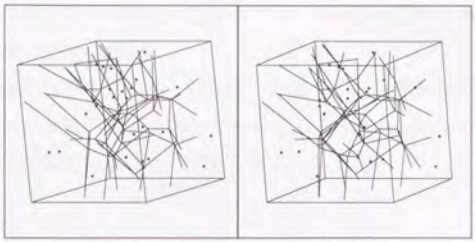


図 3.1. 3次元 Voronoi 図

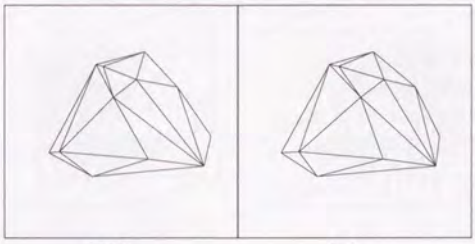


図 3.2. 3次元 Delaunay 図

て発生させた25個の点を母点とした．そして，この Voronoi 図より生成された3次元 Delaunay 図を図3.2に示す．ここでは，視認性を高めるため，凸包表面の手前側の分割のみを示してある．これ以降，計算機出力として得られた3次元 Delaunay 図はすべて同様の表示を行なうことにする．

それでは次に，立方格子上に母点を配置した場合を考えよう．一般の3次元 Voronoi 図では，同一球面上に載っている4個の母点により，一つの Voronoi 点が決定されるのであるが，母点が立方格子上に配置された場合には，8個の母点が同一球面上に載っており，退化状態になっている．立方格子上に置かれた125個の母点集合に対し，Voronoi 図を構築した結果を図3.3に示す．第2章でみたように，位相優先に基づくアルゴリズムでは退化状態に対する特別な処理を行なっていないが，このように安定した出力を得ることができる．これは，位相構造の無矛盾性を優先させるという方針に基づいているため，退化した状態であっても長さ0の辺や面積0の面が採り入れられて，退化していない Voronoi 図と同じ位相的構造として扱われているからである．

このようにして得られた Voronoi 図をもとに，その双対図形である Delaunay 図を求めてみたものが，図 3.4である．

この表面の分割の様子を見ている限りは，問題なく四面体分割がなされているように見えるが，内部の分割の様子を調べてみると， Delaunay 図としては不適切な出力になっていることがわかったので，次節で詳しくみてみる．

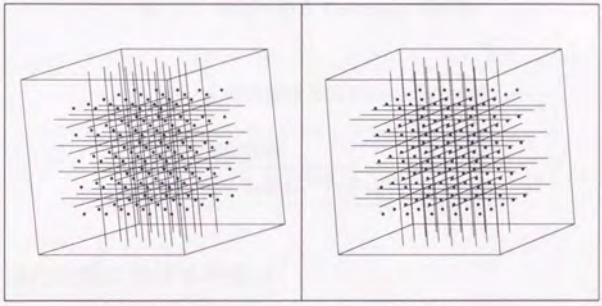


図 3.3. 立方格子上に配置した母点に対する Voronoi 図

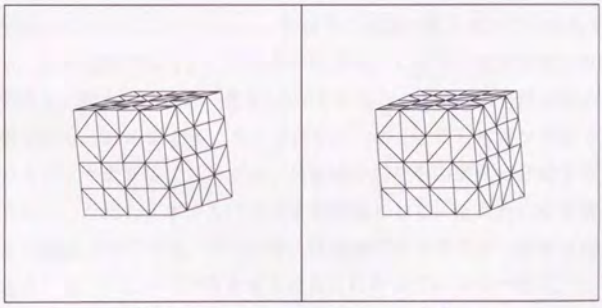


図 3.4. 立方格子上に配置した母点に対する Delaunay 図

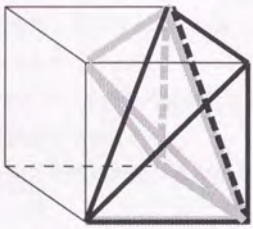


図 3.5. 重なり合う Delaunay 四面体

表 3.1. 全体の体積と四面体の合計体積

立方体全体の体積	0.21600
全 Delaunay 四面体の合計体積	0.26887

3.1.1 退化状態における問題点

立方格子上に配置した母点集合に対して出力された3次元 Delaunay 図を図3.4に示したが，この内部において，図3.5に示すような，互いに重なり合う Delaunay 四面体が生成されていた．図中の細線は格子線を表し，2種類の太線が二つの Delaunay 四面体を表している．

また，生成されたすべての Delaunay 四面体の体積の和を求めてみた結果を表3.1に示す．なお，この実験で入力として用いた母点は，1辺 0.6 の立方体の表面および内部の立方格子上に置かれている．表3.1の値をみると，四面体の合計体積が，全体の立方体の体積よりも，24% ほど多くなっており，このことからも重なりあった四面体が存在していることがわかる．これでは，凸領域の内部を四面体で分割するという目的は達成できない．ただし，本アルゴリズムの特徴として，位相的には矛盾のない出力であることは保証されている．すなわち，位相的には矛盾のない分割であるが，計量的には「重なり合う」という矛盾を含んだものになっているのである．

調べてみると，これは単なるアルゴリズムの設計ミスといった問題ではなく，退化している3次元 Delaunay 図特有の難しさに起因することが新たにわかってきた．この難しさは，理論的なアルゴリズム構築段階では発見できず，実際に計算機実験を行なうことによって初めて顕在化してきたものである．実際に3次元 Delaunay 図を利

用した四面体分割システムを構築する場合には考慮しなければならない問題点である．

前に述べたように，位相優先のアルゴリズムでは退化している場合の出力に微細構造が含まれている．この微小な乱れは，3次元 Voronoi 図においては表面化することなく問題にならなかったのであるが，その双対図形である Delaunay 図においては，大きな乱れとなって現れてしまうのである．

次節では，なぜこのような大きな乱れが生じるのかを調べる．

3.1.2 不具合発生に至る過程

新しい母点の添加に伴う Voronoi 図更新作業を Delaunay 図側から見た更新作業と照らしあわせながら見ていくことにする．

Voronoi 点は，Delaunay 図における Delaunay 四面体に対応しており，Voronoi 点を削除することは，既存の Delaunay 四面体を分割することに対応している．まず，最初の削除すべき Voronoi 点が選ばれた時点では，図 3.6 (a) に示すように，それを含む四面体（図中で上にある四面体）が新しい母点によって分割される．次に，その Voronoi 点に隣接する Voronoi 点のうちで削除すべきと判定されたものがあれば，対応する四面体（図中で下にある四面体）も，図 3.6 (b) に示すように分割される．

さらに，隣接する Voronoi 点に対して探索が行なわれ，削除される Voronoi 点集合が選ばれていくのであるが，この過程を Delaunay 図側から捉えると，削除される Voronoi 点に対応した四面体が，新しい母点を使って次々と分割されていくことになる．

さて，退化している場合には，新しく添加された母点が，既存の Delaunay 四面体の外接球上に載っているため，計算誤差を含む数値計算による内外判定では，どちらに判定されるかはランダムに決まると考えてよい．

例えば，図 3.7に示すような，立方格子上に頂点を持つ二つの四面体に対し，二重四角で表された母点が新たに添加されるとする．この場合には，二つの四面体は同一の外接球を持ち，その外接球面上に新しい母点が載っているので退化状態になっている．そのため，この二つの四面体が分割されるか否かはランダムに決まる．ここで，手前の黒い太線で示された四面体は分割されないと判定され，かつ奥の灰色の太線で示された四面体は分割されると判定されることもあり得る．しかし，その場合には，手前にある分割されない四面体を貫いた四面体分割ができてしまう．図 3.5 に示した重なり合う四面体は，まさにこの状況のもとで生じたのである．

このような場合にも，Voronoi 図においては，長さがほとんど 0 の辺や面積がほと

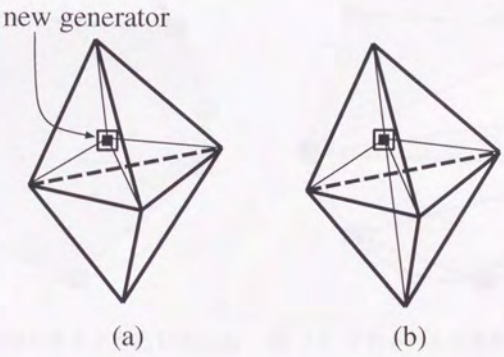


図 3.6. 新しい母点によって分割される四面体

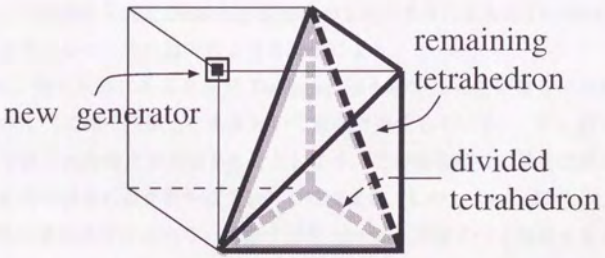


図 3.7. 分割される四面体の手前に分割されない四面体がある例

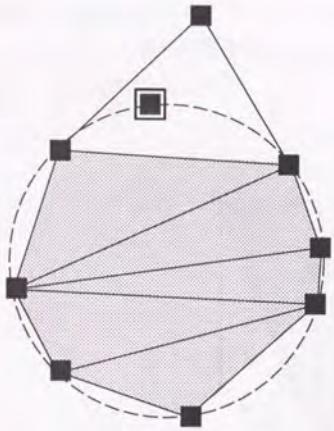


図 3.8. 退化状態にある 2 次元 Delaunay 図

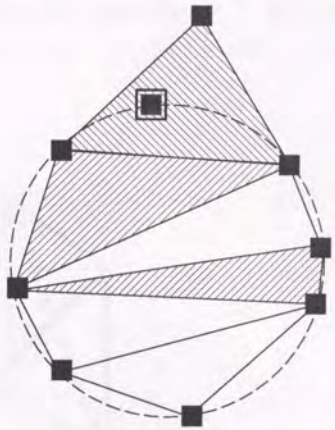


図 3.9. 分割される三角形が隣接していない状況

んど 0 の面が生じるだけなので，Voronoi 図として表示しても大きな乱れは見られない．しかし，Voronoi 図から Delaunay 図を生成するときには，Voronoi 図の辺の長さや面積に関係なく，その位相構造のみによって母点同士の接続関係が決定されるため，Delaunay 図の生成においては不具合が生じたのである．

実は，この問題は 3 次元 Delaunay 図特有のものであり，2 次元 Delaunay 図の構成においては生じない．それは次のような理由による．

図 3.8に，退化状態にある 2 次元 Delaunay 図を示す（灰色で塗りつぶされた三角形の頂点がすべて同一円周上にあるという意味で退化している）．今，同じ円周上に二重四角で表された母点が添加されたとしよう．この場合には，退化状態にある既存の三角形が再分割されるか否かはランダムに決まる．しかし，2.4.4節で示したように，Voronoi 図の構成過程において，削除すべき Voronoi 点はすべて隣接するように選ばれているために，Delaunay 図において隣接していない三角形が分割されると判定されることはない．そのため，図 3.9 に示すような状況（分割される三角形を斜線で示した），すなわち，新しく添加された母点から見て手前の三角形が残されると判定されて，奥の三角形が分割されると判定されるような状況が生じることはない．

これに対して，3 次元では状況が異なる．図 3.10において，太実線で示された四面

体が残されると判定され，そこより先には探索を進まない場合であっても，灰色で塗られた面を介して裏の四面体まで探索が及ぶことがある．このため，図3.7のような状況が生じ得るのである．

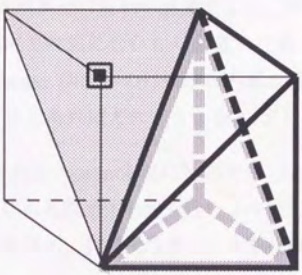


図 3.10. 不具合に至る過程

3.2 対策

3.2.1 改良点

削除すべき Voronoi 点集合の探索において，次の二つの改良を加える．

（1）削除すべき Voronoi 点集合の最初の要素として，対応する Delaunay 四面体が新たに添加された母点を内部に含むものを選ぶ．（第 2 章の 2.4.4節で述べたアルゴリズムでは，Delaunay 四面体の外接球の内部に新しい母点が含まれるという性質をもつ Voronoi 点であればどれでもよかった．）

（2）削除すべきと判定された Voronoi 点に対応する Delaunay 四面体が，添加された母点から見て手前にある四面体を回り込むようにして探索された奥の四面体であるか否かを判定するため，計量的なチェックを行なう．

3.2.2 回り込み探索の判定

“回り込み探索” である場合とそうでない場合の判定は次のような数値計算によって行なう．その様子を図 3.11に示す．

回り込み探索を判断するための手続き

（1）今判定が行なわれている四面体が，どの四面体に隣接するものとして探索されたかを調べ，その四面体との共有面を得る（図中では，この面を斜線で示してある）．

（2）四面体の頂点のうちこの共有面に含まれないものと，新たに添加された母点を結ぶ線分を作る．

（3）ステップ (1) で得られた面とステップ (2) で得られた線分が交点を持つか否かを数値計算により判定する．

（4）ステップ (3) で，交点があれば，回り込み探索になっていないと判断する．交点がなければ，回り込み探索が行なわれていると判断する（同図 (a) の場合は，回り込み探索でないと判断され，(b) の場合は，回り込み探索であると判断される）．

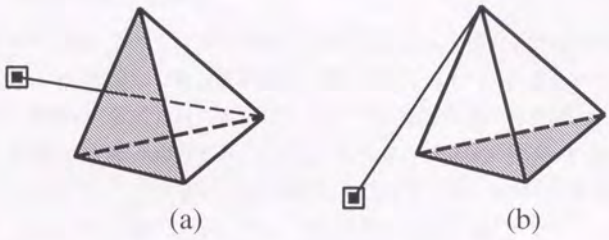


図 3.11. 回り込み探索の判断：(a) 回り込み探索でないと判断される場合；(b) 回り込み探索であると判断される場合

3.2.3 追加された処理

新しい母点の添加によって分割される四面体集合の決定の際に，位相構造に矛盾をきたすこともなく，“削除すべき”と判定された四面体であっても，さらに，回り込み探索のチェック結果に基づいた以下の処理を加えることにする．これにより，新しく添加された母点から見て手前にある四面体が残され，奥の四面体が分割されるような状況は生じなくなる．

(1) 回り込み探索でなければ，それまでの判定どおり “削除” と決定する．

(2) 回り込み探索であれば，いま判定を行なっている四面体よりも添加された母点側にある四面体の状況を調べる．

(2-a) “残す” と決定されているものがあれば，いま判定を行なっている四面体について，それまでの判定を覆し，“残す” ことに決定する．

(2-b) 現段階では “残す” と判定されたものはないが，まだ未決定のものが残っている場合には，いま判定を行なっている四面体について判定を保留し，添加された母点に近い四面体の判定を先に行なう．

(2-c) 添加された母点側にあるすべての四面体が “削除” と決定されていれば，いま判定を行なっている四面体についても判定どおり “削除” と決定する．

3.2.4 数値判定の安定化

これまでの議論とは少しはずれるが，3次元の Voronoi 図および Delaunay 図の構築において，より安定した数値計算結果を得るために加えた工夫について述べる．位相優先法に基づいて設計されたアルゴリズムでは，数値計算の精度がいかに悪くとも位相的に矛盾のない解を出力する．しかし，このことは数値計算はいい加減にやっても構わないということではない．より信頼のおける解を得るためには数値計算上の工夫も疎かにしてはいけない．ここでは，退化状態にある母点に対して，より安定した数値計算結果を得るために加えた工夫について述べる．

3.1 で取り上げた立方格子上の母点配置に対する Delaunay 図において，同一平面上にある 4 個の母点を頂点とする四面体が生成されることがある．その例を図 3.12に示す．

2.4.3 で述べたように，削除すべき Voronoi 点を選ぶためには，新母点が一つの Voronoi 点のまわりにある 4 個の母点を通る球の内部に含まれるか否かの数値判定を行なっている．この計算を，図 3.12のようなつぶれた四面体に対して行なうと，四面体の外接球が無限大（または不定）になったり，外接球の中心として決まる Voronoi 点が無限遠（または不定）になるなどの問題が生じる．

そこで，まずつぶれた四面体が作られる時の状況をみてみる．図 3.13(a) に示すように，互いに隣接する二つの Delaunay 四面体 D_1, D_2 のそれぞれの外接球の共通円周上に新母点が載っているとする．この場合には，四面体 D_1 および D_2 が分割されるか否かの数値判定はランダムに決まるため，D_1 は分割されると判定され，D_2 は残されると判定されることがある．その結果，同図 (b) で示すようなつぶれた四面体 D_3 が生成される．

これより，生成されるつぶれた四面体の 4 頂点は，常に隣接する四面体の外接球上に載っている（非常に近い）ことがわかる．そのため，次の工夫を行なうことで，つぶれた四面体に対しても安定した数値計算を行なうことができる．

つぶれた四面体に対する数値判定

4 頂点が同一平面上に載っているような Delaunay 四面体に対しては，その四面体に外接する球として，隣接する Delaunay 四面体の外接球を使うことにする．また，新しく作られる Voronoi 点には，隣接する四面体の外接球の中心を当てることにする．

例えば，図 3.13(b) において，つぶれた Delaunay 四面体 D_3 に外接する球として，

D_1 (または D_2) に外接する球を使い，対応する Voronoi 点を，D_1 (または D_2) の外接球の中心に決める．このように，同一平面上に載っている 4 点を数値計算に使わず，隣接する四面体の 4 頂点で代用することにより，数値計算の安定化を計ることができる．

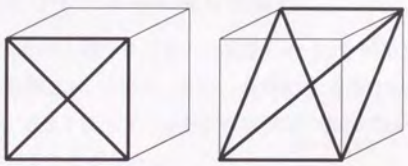


図 3.12. つぶれた四面体

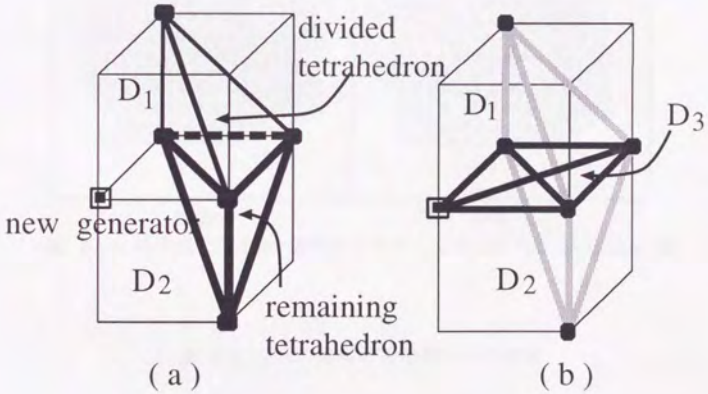


図 3.13. つぶれた四面体の生成

3.3 計算機実験

3.2 で述べた改良を組み込んだ3次元 Voronoi 図構築法を計算機に実装し，実験によりその有効性を検証した．

実験1： 母点を立方格子上に配置した場合

立方格子上（長さ 0.6 の1辺を9分割）に配置した 1000 個の母点に対して得られた3次元 Delaunay 図を図 3.14 に示す．また，全体の立方体の体積と全ての Delaunay 四面体の合計体積を，表 3.2 に示す．ふたつの体積が一致しており，重なりのない四面体分割が出力されていることがわかる．

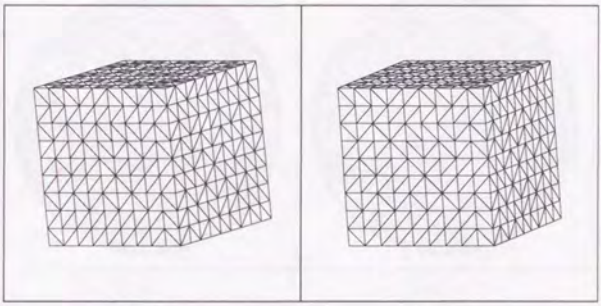


図 3.14. 格子点上の 1000 個の母点に対して出力された Delaunay 図

表 3.2. 全体の体積と四面体の合計体積

立方体全体の体積	0.21600
全 Delaunay 四面体の合計体積	0.21600

実験２： 母点を球面上に配置した場合

半径 0.5 の同一球面上にランダムに発生させた 1000 個の母点に対して出力された 3 次元 Delaunay 図を図 3.15に示す．この場合には，非常に激しい退化が生じているにもかかわらず，処理が破綻することなく出力が得られている．また，凸包の体積と全ての Delaunay 四面体の合計体積を，アルゴリズムの改良前と改良後のそれぞれの出力に対して求めてみた結果を表 3.3 に示す．改良前の出力では，四面体の体積の合計値が凸包の体積の 78 倍以上になっており，多くの Delaunay 四面体が重なりあっていることがわかる．これに対して，改良後の出力では適切な四面体分割が出力されていることがわかる．

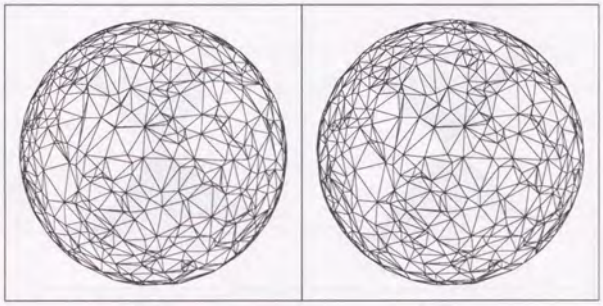


図 3.15. 球面上の 1000 個の母点に対して出力された Delaunay 図

表 3.3. 凸包の体積と四面体の合計体積

凸包の体積	0.51727
全 Delaunay 四面体の合計体積 (改良前)	40.62051
全 Delaunay 四面体の合計体積 (改良後)	0.51727

実験３： 母点を円周上に配置した場合

3.2.4 で述べた工夫による数値計算の安定性を確認するために，次のような母点データを使って実験を行なった．まず，球面の両極の上に１点ずつ配置し，続いて赤道上にランダムに 98 個の母点を発生させた．このような母点に対しては，赤道上の母点のみを使ったつぶれた Delaunay 四面体が多く生成される．出力された Voronoi 図を図 3.16に，Delaunay 図を図 3.17に示す．この出力では，特に乱れた部分もなく，つぶれた四面体に対しても安定した数値計算がなされていることがわかる．また，凸包の体積と全ての Delaunay 四面体の合計体積を表 3.4に示す．この場合においても重なりのない四面体を出力していることがわかる．

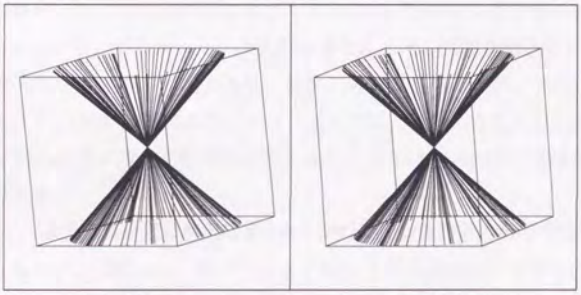


図 3.16. 赤道上に置かれた９８点と両極に置かれた２点を母点とする Voronoi 図

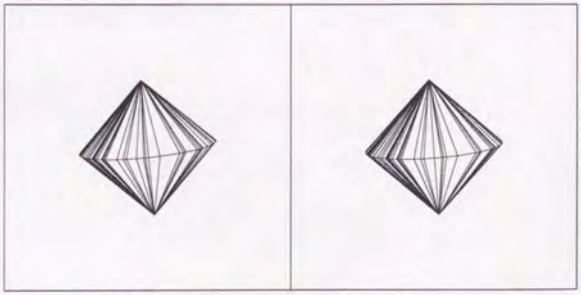


図 3.17. 赤道上に置かれた９８点と両極に置かれた２点を母点とする Delaunay 図

表 3.4. 凸包の体積と四面体の合計体積

凸包の体積	0.13360
全 Delaunay 四面体の合計体積	0.13360

3.4 考察

3次元の場合には，Voronoi 図に比べて Delaunay 図のほうが数値誤差による影響が表面化しやすいことがわかった．2次元の場合にも同じ問題が起き得るのであるが，それは位相的な探索順序の問題であり，簡単に解決できる．実際，文献 [杉原, 1991; Sugihara and Iri, 1994] で提案したアルゴリズムでは，そのような問題は起きない．逆に Voronoi 図のほうに現れた数値誤差による乱れが Delaunay 図では消えてしまうという特徴がある．

ところが，3次元の場合には状況はまったく変わり，数値誤差の影響で生じた Voronoi 図の小さな乱れが，Delaunay 図では大きく拡大されて現れる．これは，位相的な処理のみでは対処できない問題であり，計量的なチェックを加えることで対策を講じた．このことは，3次元 Delaunay 図を利用した実用的なプログラムを作ろうとする際には考慮しなければならない事実であろう．

第 4 章

制約つき Delaunay 図の近似構成法

本章では，位相優先法を適用した 3 次元 Voronoi 図構成算法が，その骨格となる位相的処理を全く修正することなく，数値判定部のみの変更によって，制約つき Delaunay 図の構成に適用できることを示す [稲垣, 杉原, 1994b]．そこでは，位相優先法の大きな特徴である「どんなに計算誤差が生じようとも位相的には矛盾のない解を出力する」という性質を積極的に利用しており，位相優先法ならではの拡張になっている．ただ，ここで紹介するアルゴリズムは計算誤差がないとしても常に真の Delaunay 図を出力するとは限らず，制約つき Delaunay 図の近似構成法であることを断っておく．

なお，本章の目的は 3 次元の制約つき Delaunay 図を構築することであるが，説明が煩雑になるのを避けるために，まずは 2 次元の例を使って提案するアルゴリズムの基本的な考え方を述べる．その後，3 次元への拡張方法について触れ，さらに計算機実験結果を見ながら 3 次元の場合のプログラムの動作を確認していく．

4.1 制約つき Delaunay 図

制約つき Delaunay 図とは，あらかじめ指定されたいくつかの辺を必ず使用しなければならないという制約のもとで一般化された Delaunay 図であり，次のように定義される [Aurenhammer, 1991; Bern and Eppstein, 1992; Okabe *et al.*, 1992]．なお，母点は一般の位置にあり，同一円周上に 4 点以上が載っているような退化した状態は考えない．

平面上の有限点集合 P と，P に属する点を端点として端点以外では互いに交わらない線分の集合を C とする．P に属する点 p, q, r において，線分 $\overline{qr}$ 上の任意の点（端

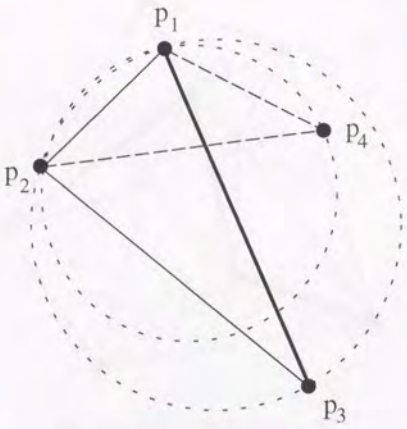


図 4.1. 制約つき Delaunay 辺

点を除く） s に対して，$\overline{ps}$ が $C-\{\overline{qr}\}$ に属するどの線分とも交わらないときに，"p は $\overline{qr}$ から見える" という．そして，$\overline{qr}$が制約つき Delaunay 図の辺として使われるのは，$\overline{qr}$が C に属する場合か，または ある p に対して，p,q,rを通る円の内部に Pに属しかつ "$\overline{qr}$から見える" 点を含まない場合である．

図 4.1 に例を示す．太線で示した $\overline{p_1p_3}$ を制約とすると，p_1,p_2,p_3 を通る円の内部には，P に属する点でかつ $\overline{p_1p_2}$から見えるものは存在しない（p_4 は円の内部にあるが，$\overline{p_1p_2}$ からは見えない）ので，$\overline{p_1p_2}$ は制約つき Delaunay 辺である．$\overline{p_2p_3}$ も同様である．また，$\overline{p_3p_1}$ は，与えられた制約であるから制約つき Delaunay 辺になる．よって，三角形 $p_1p_2p_3$ は制約つき Delaunay 図における一つの領域である．(この図の場合，もし制約がなかったとすると，三角形 $p_1p_2p_4$ が Delaunay 領域になる)．

このような制約つき Delaunay 辺によって得られた Pの凸包内部の三角形分割を制約つき Delaunay 図とよぶ．図 4.2 に制約つき Delaunay 図の例を示す．制約となる線分は太線で示してある．

図 4.2. 制約つき Delaunay 図

4.2 位相優先に基づく制約つき Delaunay 図の近似構成法

4.2.1 基本方針

2.4.3節で述べたように，新しい母点の添加に伴う Voronoi 図の更新において，削除されるべき Voronoi 点集合を求めるために数値判定を利用している．この数値判定結果をねつ造することで，制約となっている母点が隣接しているような位相構造を意図的に出力させようというのが基本的な方針である．当然，その結果得られる出力は正しい Voronoi 図からははずれたものになっているが，平面グラフであるという位相的性質は満たされているため，その双対図形を作ることはできる．そして，それが指定された線分をすべて使った制約つき Delaunay 図になっているのである．

簡単な例を使ってこの様子を見てみよう．図 4.3に，12 個の母点（黒丸）とそれに対する Voronoi 図（破線）および Delaunay 図（実線）を示す．この母点配置において，p_i, p_j を結ぶ線分を必ず使うように制約をつけた場合を考えよう．制約をつけない場合には，線分 $\overline{p_i p_j}$ は分割に使われていない．そこで，Voronoi 図の構築過程において，数値判定をねつ造することで，母点 p_i と p_j が隣接するような位相構造が作られるようにする．もちろん，制約の影響を受けないところでは，これまで通り数値計算した値を利用する．この結果得られた出力を図 4.4に示す．数値判定をねつ造したために中央

図 4.3. 制約なしの Voronoi 図と Delaunay 図

付近で乱れが生じているが，得られた位相構造に従って双対をとってみると，図 4.5に示した制約つき Delaunay 図が得られる．

このような方針が可能であるのは，位相優先法を適用して設計されたアルゴリズムにおいては，Voronoi 点の座標値などの計量的なデータは，位相構造から決まる二次的なデータとして扱っているので，その値が正しいものでなくとも，処理が破綻することなく，位相的には矛盾のない結果を出力するためである．ここで提案するアルゴリズムにおいては，この性質を積極的に利用し，意図的に数値判定をだますことで，ほしい位相構造を出力させている．

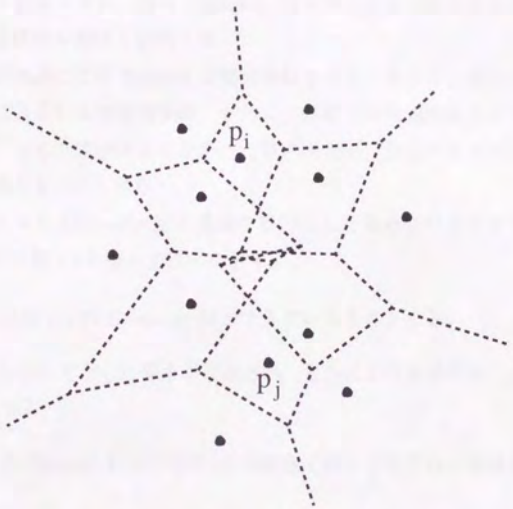


図 4.4. ねじれた Voronoi 図

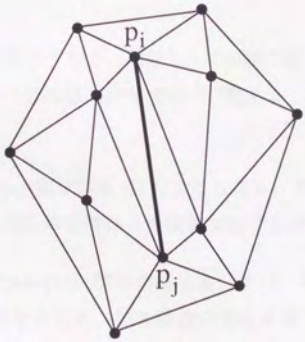


図 4.5. 得られた制約つき Delaunay 図

4.2.2 Delaunay 三角形が得られない場合

しかし，上記の基本方針に従ってアルゴリズムを構築した場合，たとえ計算誤差による判断の誤りがなくても，真の Delaunay 図が得られない場合がある．そのような状況について具体例を挙げて説明する．

まず，逐次添加法による Voronoi 図構成過程を考えてみると，新しい母点の添加に伴なって新たに生まれる隣接関係は，すべてこの新しい母点が関係しているものである．すなわち，今まで隣接関係になかった母点同士が，新しい母点の添加に伴なって新たに隣接関係をもつことはない．

ところが制約つき Delaunay 図の構築を目的とした場合には必ずそうなるとはいえない．その状況を図 4.6を使って以下に示す．

- 今までに同図 (a) の Delaunay 図ができているものとする．
- 同図 (b) に白丸で示した母点が添加され，これにより太線で示した制約が新たに加わるとする．
- 新しい母点の添加によって同図 (c) の灰色で塗りつぶされた領域が再分割の対象になる．
- そこで，新たに添加された母点を使って同図 (d) に示したような新しい分割が得られる．
- しかし，実はこの分割のうちで，同図 (e) に斜線で示した領域は Delaunay 三角形にはなっていない（外接球の内部でかつ “見える” ところに別の母点を含んでいる）．
- 真の制約つき Delaunay 図は同図 (f) のようになる．すなわち，隣接関係になかった古い母点同士に新たな隣接関係（太破線で示す）が生まれるのである．

このように，制約つき Delaunay 図の構築においては，母点添加時の位相構造の変化が制約のない場合とは異なるため，位相構造の変化を扱う骨格となる処理を共有して，数値判定結果のねつ造だけで対応するのは無理がある．

しかし，実用的には，すべての領域が正確に Delaunay 領域になっていなくても構わない場合が多い．制約を満たした上で真の Delaunay 図に近いものが得られれば利用価値があろう．大切なのは本アルゴリズムが数値的に非常に安定していることである．

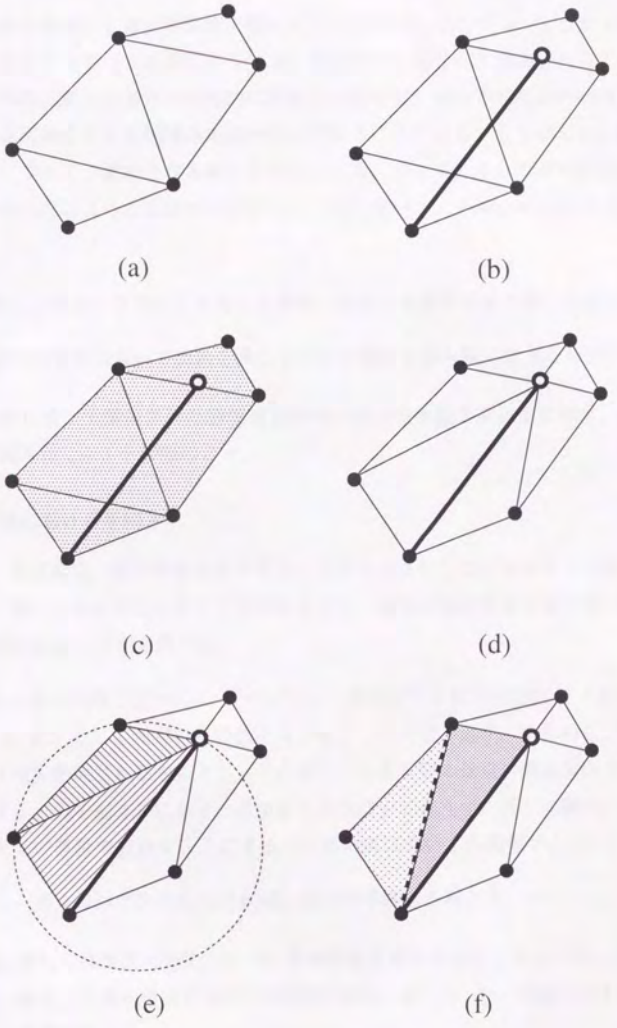


図 4.6. Delaunay 三角形が得られない状況

4.2.3 数値判定のねつ造手続き

制約のある場合にも母点は任意の順序で添加されるものとする．したがって，制約線分の端点となっている母点とそうでない母点が入り混じって添加される．ここで注意したいのは，新しい母点の添加時に制約となるのは，指定された制約のうち，その時点までに両端点とも添加済みの線分のみであることである．したがって，母点の添加が進むにつれて，制約となる線分も増えてくる．そして，それらの制約を満たすデロネー分割になるように数値判定結果をねつ造していく．そのために次の二つの方針をとる．

方針1 新しい母点の添加により新たに制約に加わった線分を必ず使う分割にする．

方針2 既存の分割において制約を満たしている線分を取り除くような分割にしない．

この方針に沿って構築された数値判定結果のねつ造手続きを以下に示す．本来の数値判定手続きは 2.4.3 を参照のこと．

数値判定部における手続き

1. 今，数値判定（取り除かれるか否か）を行なおうとしているボロノイ点に隣接する3個の母点を頂点とする三角形の各辺と，新たに制約となる線分が，端点以外で交わるかどうかを調べる．

2. もし，端点以外で交われば，そのボロノイ点に対する数値判定値 H（第2章2.4.3節 2.1 式）として任意の負の値をねつ造し，その値を返す．すなわち，そのボロノイ点を削除されるものとし，その回りにある3個の母点の隣接関係を壊すようにする．例を図 4.7 に示す．添加された母点を二重丸で，新たに制約となった線分を太い黒線で表わすことにする（以降の図においても同様である）．

3. もし，端点以外で交わらなければ，以下の手続きを行なう．

   (a) 新しく添加された母点が，今，数値判定を行なおうとしているボロノイ点に隣接する3個の母点を通る円の外部にある（$H \geq 0$）か，内部にある（$H < 0$）を調べる．

   (b) もし，円の外部にある（$H \geq 0$）と判定されれば，行列式 Hの値をそのまま返す．すなわち，その Voronoi 点は “残される” と判定する．

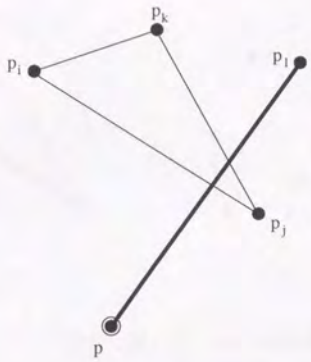


図 4.7. 新たに制約となった線分と交差する場合

(c) もし，円の内部にある（$H < 0$）と判定された場合には，次のように判定する．

i. 新しく添加された母点から，円周上の3個の母点に対して引いた線分が，その時点までに制約となっている線分と，端点以外で交差するかどうか調べる．

ii. もし，交差すれば，そのボロノイ点に対する数値判定値として，任意の正の値をねつ造し，その値を返す．すなわち，そのボロノイ点は“残される”と判定し，その回りにある3個の母点の隣接関係を残すようにする．例を図4.8に示す．既に制約となっていた線分を太い灰色の線で表す（図4.9も同様である）．

iii. もし，交差しなければ，行列式Hの値（負の値）をそのまま返す．すなわち，そのVoronoi点を“削除する”と判定する．例を図4.9に示す．

上の手続きにおいて，数値判定結果をねつ造しているのは，ステップ2と3(c)iiである．前者は，方針1に基づいて行なわれたものであり，後者は，方針2に基づいて行なわれたものである．

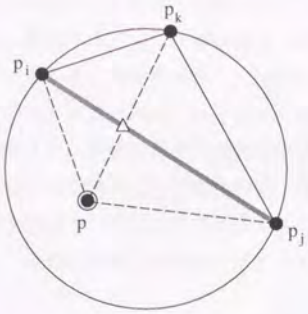


図 4.8. すでに制約となっていた線分と交差する場合

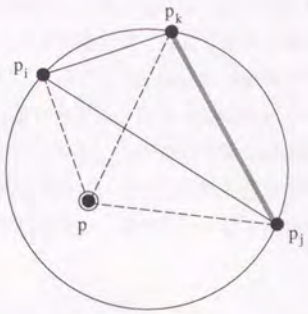


図 4.9. すでに制約となっていた線分と交差しない場合

4.2.4 制約を満たす工夫

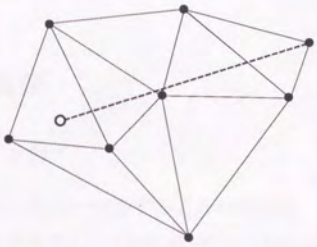
制約つき Delaunay 図の位相構造としては，通常の Delaunay 図の位相構造を満たすだけでなく，得られた分割が制約として与えた線分をすべて使っていなければならない．

しかし，通常の Delaunay 図構成法に基づき，数値判定部分の変更のみで対応させたアルゴリズムでは，計算誤差の影響により上の要求を必ず満たすとは限らない．

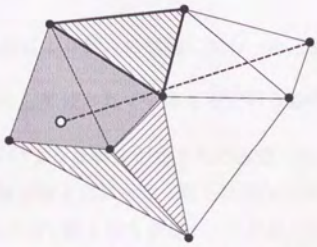
例えば，図 4.10 (a) において，白丸で示された母点が新たに添加され，点線で表された制約が加わったとしよう．これに伴ない再分割される領域をすべて求めるために隣接する領域を次々とたどっていくのであるが，計算誤差のために判断を誤り，同図 (b) に示すような判定がなされる可能性もある．ここで灰色に塗られた三角形が分割されると判定されたものであり，斜線の三角形は残されると判定されたものであり，境界が太線で表されている三角形に対する判断を誤っている．そのため，探索がこれ以上先に進まず，新たに制約となった母点対が隣接しない．

そこで，Delaunay 図の位相構造に矛盾がないことに加え，制約となっているもう一方の母点に到達するような位相構造を得ることを，数値判定結果よりも優先させる必要がある．すなわち，位相的に矛盾がなくかつ数値計算の結果これ以上分割領域は広がらないという判定がなされても，新たに制約となったもう一方の母点に到達して

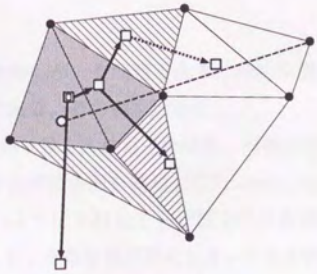
いなければさらに探索を進めることにする．その際には，最も確からしい方向から優先して探索するために，到達点となる母点までの距離が最小になるような枝を選ぶようにする．その様子を同図 (c) に示す．ここでは，対応する Voronoi 点（四角で示されている）が書き加えられており，図において二重四角で表された Voronoi 点から探索が始まり，探索結果が実線矢印で示されている．しかし，この時点ではまだ制約となる母点対が隣接していないので，数値判定結果に従わず，さらに点線矢印の方向に探索を進める．平面グラフ構造をもつ幾何図形であるため，最終的には必ず目的地までたどり着く経路が見つかるはずである．



(a) 母点添加時の状況



(b) 制約条件を満たさない場合



(c) 探索の継続

図 4.10. 制約条件を満たす解が得られない状況とその対策

4.3 3次元への拡張

4.3.1 数値判定のねつ造

前節の基本方針は自然に3次元へ拡張することができる．3次元の場合には，いくつかの指定した三角面を必ず使うという制約がついたDelaunay四面体分割を構成することになるが，2.4節で述べた3次元Voronoi図構成アルゴリズムの数値判定部分に2次元の場合と同様な修正を加えればよい．

チェックすべき条件は，以下の二つである．

[方針1] 新しい母点の添加により新たに制約に加わった三角面が使われていること．

[方針2] 既に制約三角面として使われているものが取り除かれないこと．

これらの条件を満たさないような判定がなされた場合には，数値判定値をねつ造する．[方針1]を満たさない例を図4.11に示す（新たに添加された母点を二重丸で，制約面を灰色で塗りつぶして表わす; 図4.12も同様）．これは，2次元の場合の図4.7に相当している．さらに，[方針2]を満たさない例を図4.12に示す．これは，2次元の場合の図4.8に相当している．

4.3.2 問題点

ただ，3次元の場合にはやっかいな問題もある．制約を満たすような四面体分割が存在しないことがあるのである．その例を示そう．

図4.13(a)に示すような折り目を入れた三角柱を，内側に折れる向きにひねったものが同図(b)である．この多面体の各面を制約としたDelaunay図の構成を考えてみる．すると，多面体内部をどのように分割しても分割でできる領域が多面体の外部にはみだしてしまい，制約を満たすような分割が存在しないことがわかる（一方，2次元の場合には，どのような多角形に対してもその内部を三角形分割することが可能である）．

ところが，このように制約を満たす四面体分割が存在しない場合においても，位相優先法に基づいて設計されたプログラムを使うともっともらしく解を出力する．それ

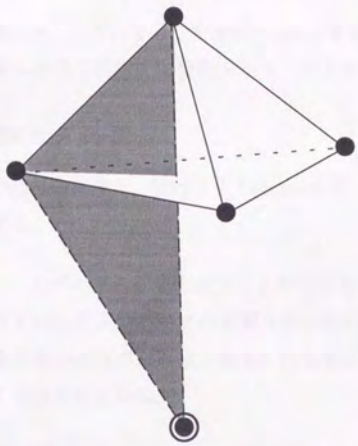


図 4.11. 新たに制約となった三角面と交差する場合

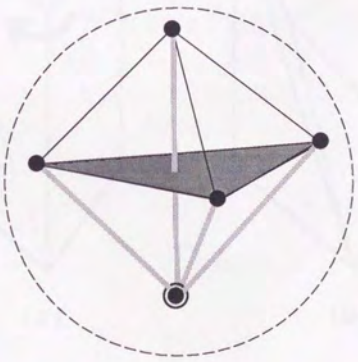


図 4.12. 制約となっている三角面と交差する場合

は位相的には矛盾のないものではあるが，現実にはまったくその解は使えない．プログラムを利用する立場からすると，少なくとも何らかのメッセージを出してほしいところである．

そこで，出力された解に対して次のような計量的チェックを行なうことで，その解の妥当性を調べ，その結果に応じて利用者に警告メッセージを出すことにする．

解の妥当性を調べる計量的なチェック

与えられた母点集合の凸包の体積と，制約つき Delaunay 図における全 Delaunay 四面体の体積の和を比較する．

これにより，利用者に入力データの変更を促すことができる．本当は，母点の追加，母点の移動，制約の変更といった入力データの変更を利用者の希望に応じてシステムが自動的に行ない，再度分割が実行されれば一番良いのであるが，これは簡単に解決できそうもなく課題として残されている．

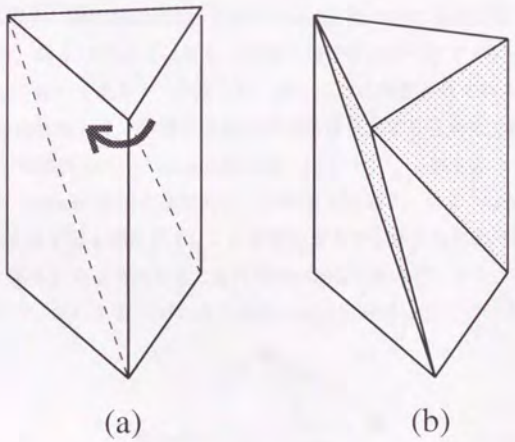


図 4.13. 制約条件を満たす分割が存在しない多面体

4.4 計算機実験

4.2, 4.3節で述べた手法を，3次元の制約つき Delaunay 図構成のためのプログラムとして計算機上に実装した．計算機実験結果をもとにプログラムの振舞いをみていく．

ただし，4.2.4節に示した工夫は未実装であり，常に制約を満たす出力が得られる保証はない．しかし，ほとんどの実験で制約を満たす出力が得られることがわかり，現段階のプログラムでも十分に実用的であると思われる．

4.4.1 動作確認

まず，図 4.14に示すような六面体の頂点上に5個の母点を配置する．制約をつけずに3次元 Voronoi 図を求めた結果を，図 4.15に示す．そして，この結果をもとに生成された Delaunay 図を，図 4.16に示す．この Delaunay 図の与える四面体分割をわかりやすく図示すると，図 4.17のようになる（分割の様子をわかりやすくするために，各四面体を少し離して描いてある）．これより，縦に三つに分割されていることがわかる．

次に，同じ母点に対して，中段の3個の母点を頂点とする三角面を制約としてみる（図 4.18）．この結果得られた Voronoi 図を図 4.19 に示す．これをみると，制約をつけたために生じる Voronoi 領域の反転のようすがよくわかる．この Voronoi 図から生成された Delaunay 図を図 4.20に示す．この分割のようすをさきほどと同様にわかりやすく図示すると，図 4.21のようになる．制約のない場合と異なり，上下二つの四面体として分割されており，制約となっている三角面を使った分割になっていることがわかる．

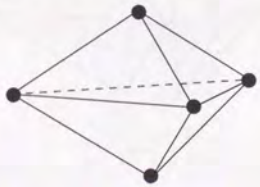


図 4.14. 母点配置（制約なし）

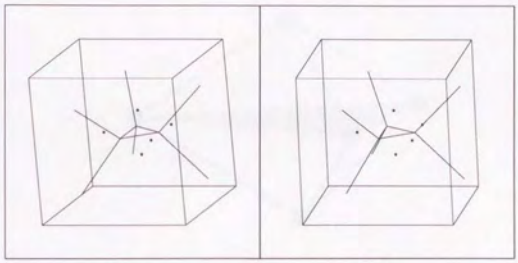


図 4.15. 制約をつけなかったときの Voronoi 図

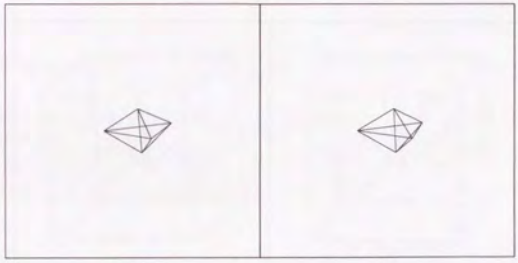


図 4.16. 制約をつけなかったときの Delaunay 図

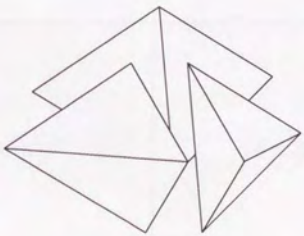


図 4.17. 制約をつけなかったときの分割のようす

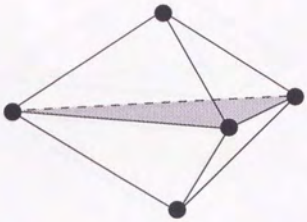


図 4.18. 制約となる面

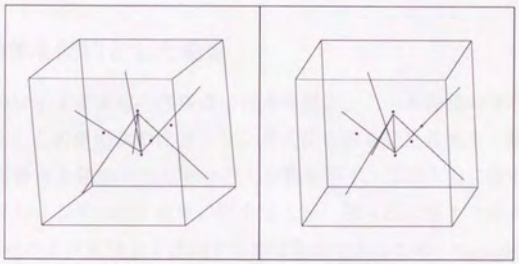


図 4.19. 制約をつけたときの Voronoi 図

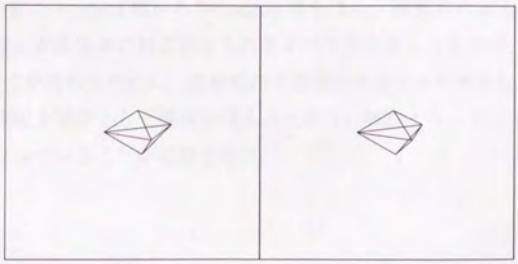


図 4.20. 制約をつけたときの Delaunay 図

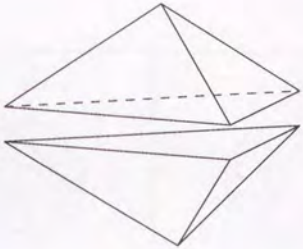


図 4.21. 制約をつけたときの分割のようす

4.4.2 多面体を制約とした場合

まず，図 4.22に示す多面体の各頂点に母点を置き，この多面体のすべての面を制約とする．さらに，この多面体の内外にランダムに母点を発生させる．図 4.23に，ランダムに 100 個の母点を発生させた場合の入力例を示す．この入力に対して，出力された Voronoi 図における Voronoi 領域の例を図 4.24，図 4.25に示す（母点が多くなると Voronoi 図からその 3 次元構造を認識するのは困難であるため，Voronoi 図全体の出力結果は載せない）．図 4.24の Voronoi 領域は，多面体の面の制約を受けたために反転している部分がある．一方，図 4.25の Voronoi 領域は，制約となっている面から比較的遠くにあり，その影響を受けていない．

ここで，得られた Voronoi 図から Delaunay 図を作り，生成された Delaunay 四面体のうち，その重心が多面体の外にあるものをすべて取り去ったものが，図 4.26である（ただし，視認性を高めるために，多面体の手前側の表面のみを表示した）．このように，多面体の表面を境界とした領域が残ることから，制約となっている面がすべて使われた分割になっていることが確認できた．

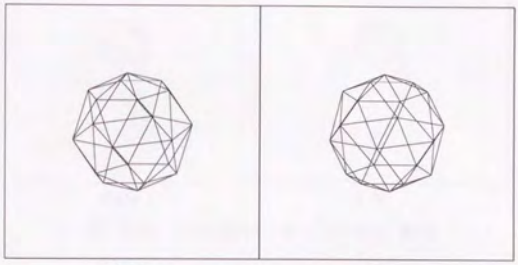


図 4.22. 制約となる多面体

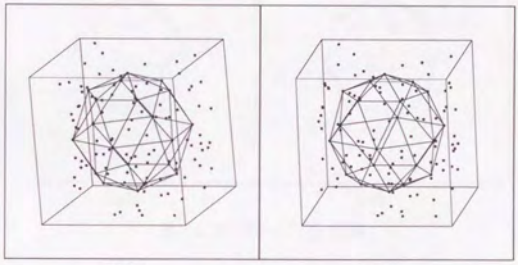


図 4.23. 入力データ

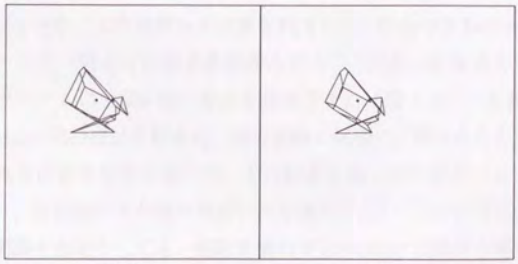


図 4.24. 反転部分を含む Voronoi 領域

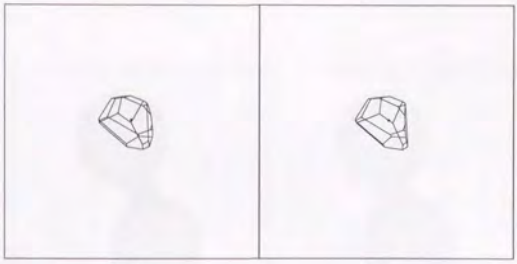


図 4.25. 反転部分のない Voronoi 領域

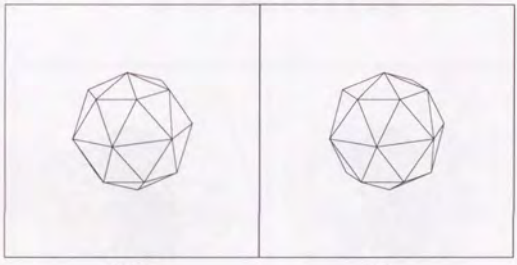
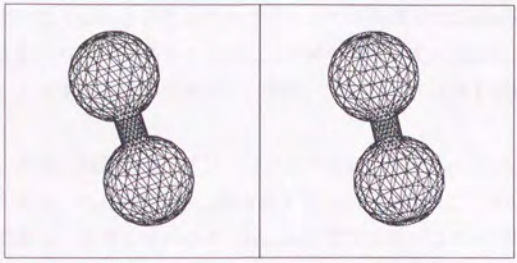


図 4.26. 残った多面体

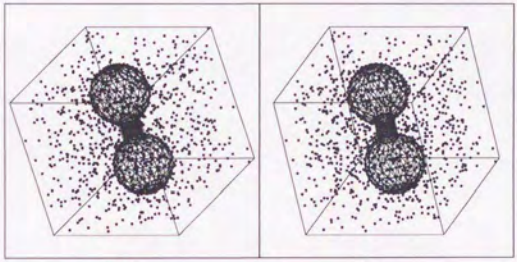
4.4.3 より複雑な多面体を制約とした場合

次に，凹形状を含む，より複雑な多面体を利用して，制約つき Delaunay 図の生成実験を行なう．そこで，図 4.27に示す多面体のすべての頂点上に母点を置き，この多面体の内外に，ランダムに 1,500 個の母点を発生させた（図 4.28）．これを入力として，制約つき Delaunay 図の構成を試みた．前の実験と同様に，得られた出力をもとに，多面体の外部にある四面体を取り除いた．その結果を図 4.29に示す（ここでも，視認性を高めるために，多面体の手前側の表面のみを表示する）．このように，比較的複雑な形状を持つ多面体を制約としても，制約を満たす Delaunay 四面体分割が得られることが確認できた．



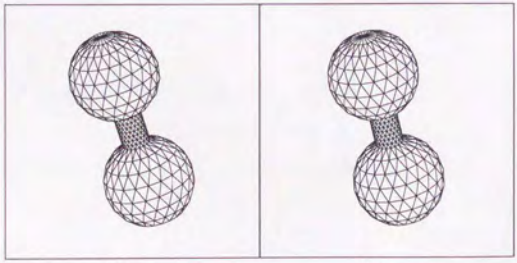
Right Left

図 4.27. 制約となる多面体



Right Left

図 4.28. 入力データ



Right Left

図 4.29. 残った多面体

4.5 考察

通常の逐次添加型 Voronoi 図構成アルゴリズムの骨格部分である位相構造の更新手続きをそのまま使って，制約つきの Delaunay 図の近似構成に成功したのは，位相優先法がもたらした予想外の効果であった．実際，計算機への実装も短期間で終えることができた．

有限要素法，補間，経路計画，グラフィックスなど，Voronoi 図／ Delaunay 図の応用分野を考えると，ある指定された面を必ず使った分割がほしいという場面は多い．そこでは多くの場合，正確な制約つき Delaunay 図が必要ではないので，本章で示した近似構成法は有効な手段を提供するといえる．

第 5 章

3 次元メッシュ生成への応用

5.1 有限要素法における 3 次元メッシュ生成

有限要素法は，構造解析・流体解析などの現場で広く利用されている手法である．計算機の能力向上に伴い，解析工程のために要する時間は短縮されてきたが，その前処理であるメッシュ生成工程にはいまだ多くの労力をかけている．そのため，この工程の自動化を目指して，メッシュ生成アルゴリズムの研究が盛んに行なわれてきた [Ho-le, 1988; 名取, 野寺, 1989; Bern and Eppstein, 1992]．しかし，大量の図形データを扱わなければならないことと，適切な要素形状への分割が難しいことなどにより，満足のいくシステムはできていない．現状では，オペレータが計算機システムと対話形式で作業を進めていることが多く，多くの時間を費やしてメッシュ生成を行なっている．特に 3 次元の有限要素解析におけるメッシュ生成作業は，形状や分割の様子が認識しにくいこともあり，オペレータに多大な負荷を与えている．そのため，メッシュ生成の自動化に対する期待が大きい．

メッシュには，格子が規則正しく並んでいる構造格子と規則性を要求しない非構造格子がある．従来，メッシュ生成の容易さから構造格子を使う方法がよく利用されてきたが，複雑な形状への対応は難しい．一方，非構造格子は格子の並びに制約がないため自由度が大きく，複雑な 3 次元形状にも対応しやすいが，その自由度の大きいことがかえって格子の生成を難しくしているともいえる．この非構造格子を生成するための手法として，Delaunay 図を使った方法が注目されている [Cavendish, *et al.*, 1985; 谷口, 太田, 1991; 藤本, 大野, 1992; Dey, *et al.*, 1992]．Delaunay 図を利用すれば，望みの密度に応じて領域内に母点を配置することにより，分割の密度をたやすく調整す

ることができる．

しかし，従来のアルゴリズムでは数値誤差の影響を受けやすく，実用に耐え得るシステムを構築できない．そのため計算誤差対策を含んだメッシュ生成法も提案されており，メッシュの位相構造をチェックしながら処理を進める方法 [Dey, et al., 1992]，誤差解析により数値判定の誤りを修正する方法 [秋山, 他, 1993] などが報告されている．

本章では，位相優先法に基づく Delaunay 図構成算法を四面体メッシュ生成問題へ適用した事例を報告する．

5.2 メッシュ生成における数値的安定性

実際の解析現場においては，効率良く計算を行なうために母点密度は必ずしも一様ではない．変化が激しいところには非常に多くの母点を，そうでないところは極力少ない母点を配置するのが望ましい．このような状況においては計算誤差の影響が大きく，誤差対策が施されていないシステムでは安定した出力が得られない．

本実験では，入力として大きな密度差のある母点集合を与え，3 次元 Delaunay 図の生成を行なった．ここでは実験例を二つ示すが，一方は入力として与える母点を規則的に発生させた場合で，もう一方は乱数を使って発生させた場合である．

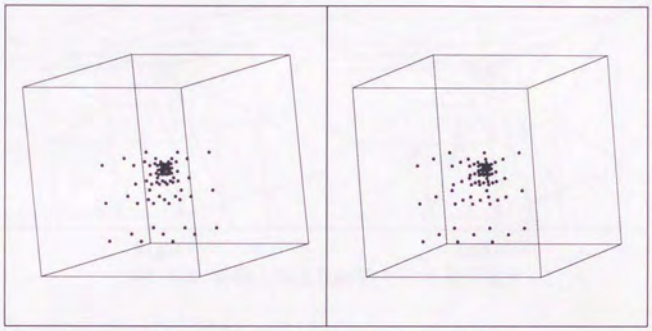
5.2.1 規則的に母点を発生

まずは，以下のようにして密度差の大きい母点集合を規則的に発生させる．

- 入力データ作成の容易さから対象物体の形状は立方体とした．
- 立方体を 8 等分しそれぞれの小立方体の頂点に母点を配置する．さらにそのうちのひとつの小立方体に対し同じ操作を行なう．これを再帰的に 1 5 回繰り返す．この操作により，隣接する母点間距離がおよそ 16,000 倍ほど差がある母点配置ができる．
- 母点数の合計は 293 個になる．

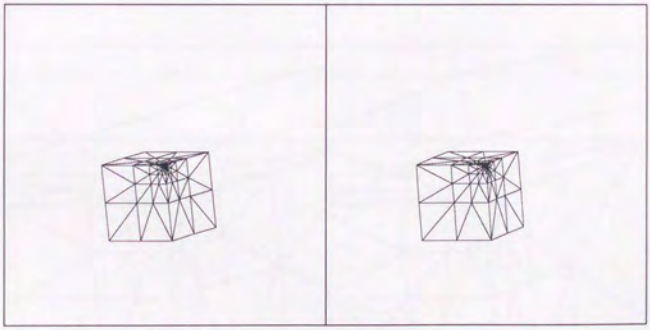
図 5.1に母点配置の様子を示す．外側にある立方体は入力として与える母点領域の境界を表している．

この入力に対して出力された3次元 Delaunay 図を図5.2に示す．ここで密度の高い部分の様子を見るためにその部分を拡大してみる．1000倍に拡大したものが図5.3で，10,000倍に拡大したものが図5.4である．



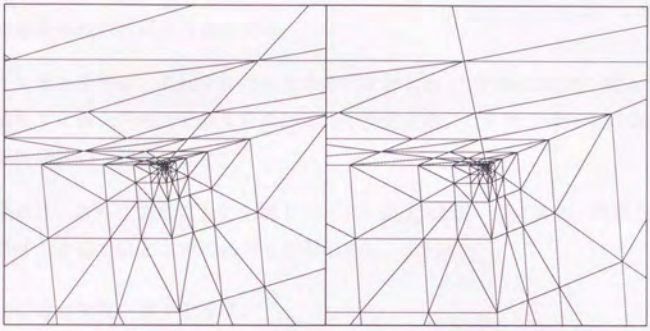
Right Left

図 5.1. 実験1で使った母点データ



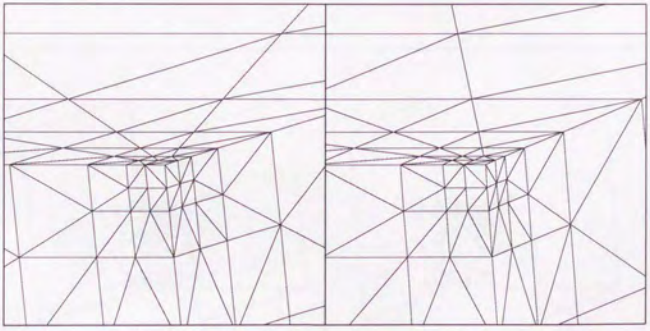
Right Left

図 5.2. 実験1の出力結果



Right Left

図 5.3. 実験 1 の出力結果（1000 倍に拡大）



Right Left

図 5.4. 実験 1 の出力結果（10000 倍に拡大）

5.2.2 ランダムに母点を発生

次に，以下のようにして密度差の大きい母点集合を乱数を使って発生させる．

- 対象物体の形状は立方体とする．
- 立方体を8等分しそれぞれの小立方体の表面上および内部に乱数を使って母点を発生させる．さらにそのうちのひとつの小立方体に対し同じ操作を行なう．これを再帰的に5回繰り返す．
- 母点は，各小立方体ごとに，面上120個・辺上48個・頂点8個・内部50個発生させ，全体では851個の母点を配置した．

図5.5に母点配置の様子を示す．

この入力に対して出力された3次元 Delaunay 図を図5.6に示す．ここで密度の高い部分の様子を見るためにその部分を拡大してみる．20倍に拡大したものが図5.7で，200倍に拡大したものが図5.8である．

これらの実験結果より，非常に大きな密度差があっても途中で処理が破綻することなく，四面体メッシュが生成されていることが確認される．

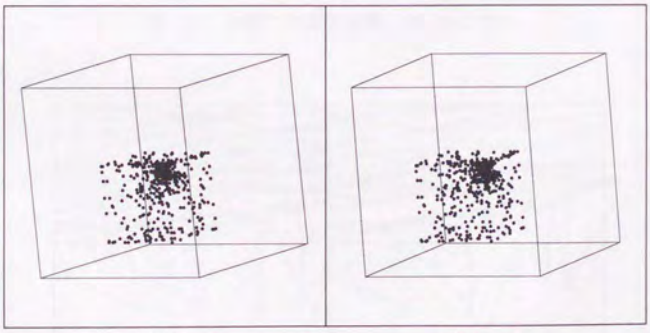


図 5.5. 実験2で使った母点データ

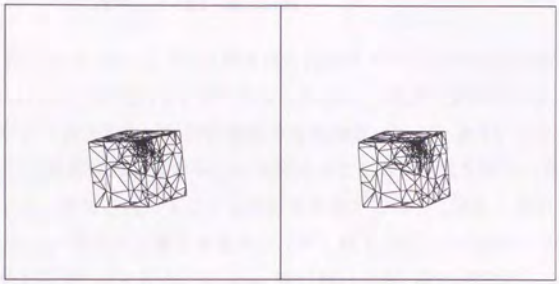


図 5.6. 実験 2 の出力結果

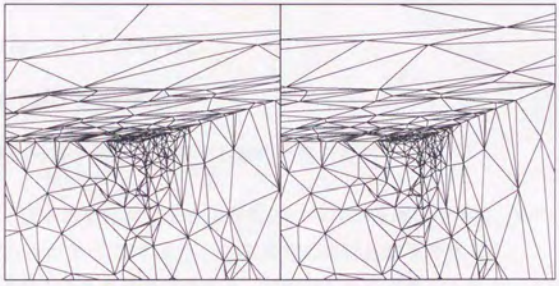


図 5.7. 実験 2 の出力結果（20 倍に拡大）

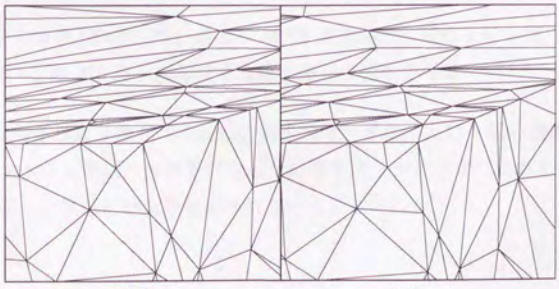


図 5.8. 実験 2 の出力結果（200 倍に拡大）

5.3 メッシュ形状の評価方法

ここで，Delaunay 図によって生成された四面体メッシュの形状を調べてみる．

有限要素法において精度のよい解を得るためには，“細長い領域がなるべくできない”ような分割がよいとされる（解析の種類や対象物などによりそういえない場合もあるが）．そこで，“細長い四面体や平たい四面体がなるべく生成されない分割”が良い分割であるとして，生成されたメッシュ形状を評価することにする．なお，2次元の場合には，Delaunay 図のもつ最小角最大性（第1章 1.2節）より細長い三角形が生成されにくいのであるが，3次元の Delaunay 図においては，状況が異なり，面と面の交角が小さいものもできてしまうことに注意する必要がある．

ここでは，四面体メッシュの形状を評価するために次の三つの指標を用いることにした．

- 三角形面における辺の内角
- 四面体における面の内角
- 四面体に外接する球と内接する球の半径の比

ちなみに，理想的な形状である正四面体における各指標の値は，辺の内角がすべて 60 度，面の内角がすべて約 70 度，内外接球の半径比が 3 である．

まずは，次のような条件で乱数を使って作成した母点集合を与え，3次元 Delaunay 図を利用した四面体分割を行なった．

- 対象物体の形状は立方体とした．
- 面上に 120 個 (20 個／面), 辺上に 48 個 (4 個／辺), 頂点上に 8 個, さらに内部に 100 個（合計 276 個）の母点を乱数を使って発生させる．

図 5.9に出力された3次元 Delaunay 図を示す．また，上の3個の指標に基づいて，生成されたメッシュ形状の分析を行なった結果を図 5.10に示す．これを見ると，細長い領域やつぶれた領域がかなり生成されていることがわかる．

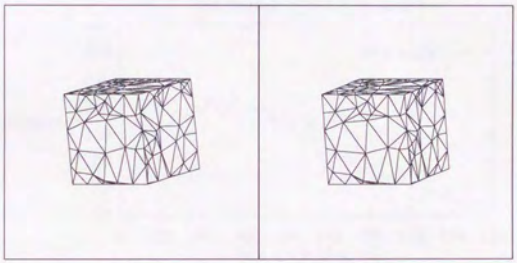
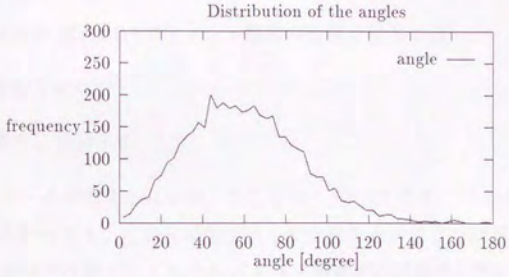
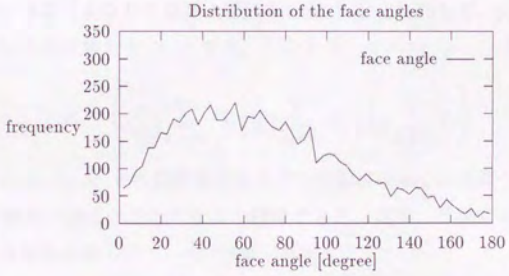


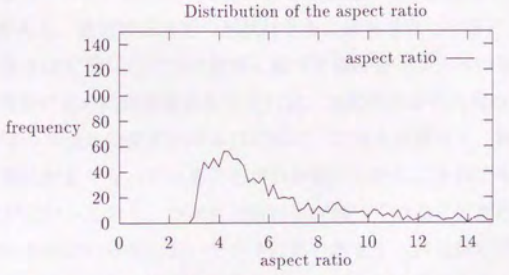
図 5.9. 初期母点配置に対する出力結果



(a) 辺の内角の分布



(b) 面の内角の分布



(c) 外接球と内接球の半径比の分布

図 5.10. 初期母点配置に対するメッシュ形状の分析

5.4 メッシュ形状の改善

3次元 Delaunay 図においてメッシュ形状の改善を計るには，

- 母点を移動させる
- 母点を新たに追加する

といったアプローチが考えられるが，ここでは，“母点の移動” による改善を試みる．

母点の移動といっても，どこに移動させるかを決めるのはそう簡単なことではない．ある四面体の形状が改善されてもそれによって隣接する四面体の形状が悪くなってしまってはしょうがない．次に示す方法は，Delaunay 図の位相構造を使うことで簡単に実現できる．

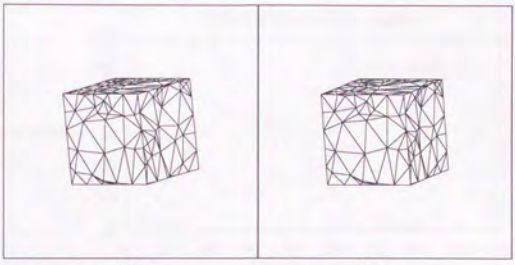
領域の内部の母点（すなわち境界上ではない母点） p_i に対して，p_i と Delaunay 辺でつながれている母点集合を A_i とする．このとき，$p_i = (x_i, y_i, z_i)$ を

$$p_i = \left(\frac{1}{|A_i|} \sum_{p_j \in A_i} x_j, \frac{1}{|A_i|} \sum_{p_j \in A_i} y_j, \frac{1}{|A_i|} \sum_{p_j \in A_i} z_j \right)$$

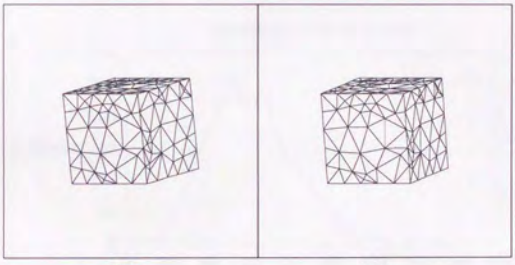
へ移す．すなわち，p_i を それに隣接するすべての母点の重心の位置へ移す．これにより全体として細長い領域が少なくなると期待できる．実際，2次元の Delaunay 図に対しては良好な結果が得られている [杉原, 1994]．

実験結果を図 5.11に示す．移動前の母点に対する Delaunay 図 （同図 (a)）に比べると，重心への移動を行なった母点に対する Delaunay 図（同図 (b)）は明らかに表面の三角形の形状が改善されているのがわかる．同じ移動操作を再度適用した後の Delaunay 図が同図 (c) である．表面の三角形の形状はさらに改善されていることがわかる．

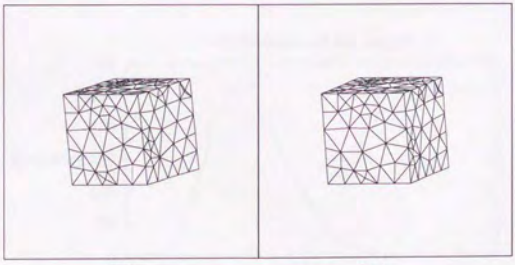
それでは，さきほど示した三つの指標に基づき四面体メッシュの形状の変化を分析してみる．三角形の辺の内角の変化を図 5.12に，四面体の面の内角の変化を図 5.13に，外接球と内接球の半径比の変化を図 5.14に示す．これらを見ると，移動前に比べ明らかにメッシュ形状が良くなっていることがわかる．しかし，それでもなお，面の内角の分布は裾野が広がっており，つぶれた四面体が残っていることがわかる．そのことは，内外接球の半径比の分布において非常に高い値をもつ四面体が存在していることにも現れている．この傾向は，移動操作をさらに繰り返し適用しても変わることはなかった．



(a) 初期母点配置に対する出力結果

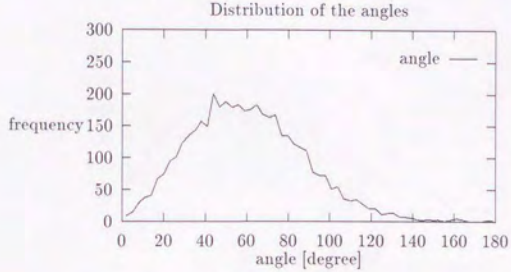


(b) 移動操作を 1 回行なった母点に対する出力結果

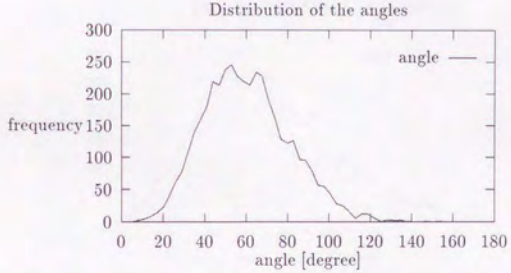


(c) 移動操作を 2 回行なった母点に対する出力結果

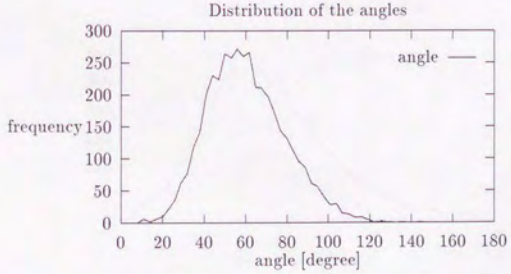
図 5.11. 重心位置へ母点の移動させたときの Delaunay 図



(a) 移動操作前

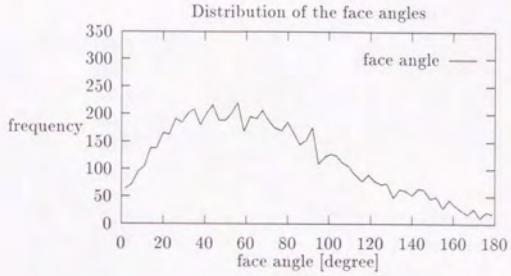


(b) 母点移動操作 1 回

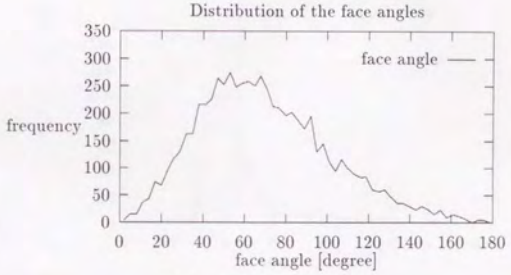


(c) 母点移動操作 2 回

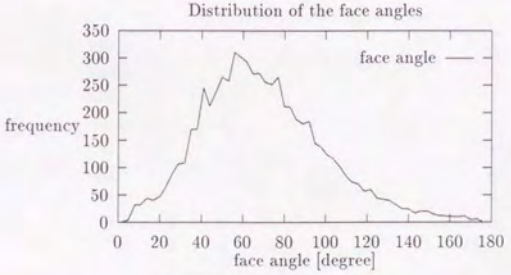
図 5.12. 三角形の辺の内角の分布



(a) 移動操作前

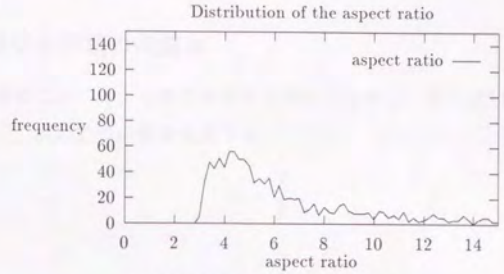


(b) 母点移動操作 1 回

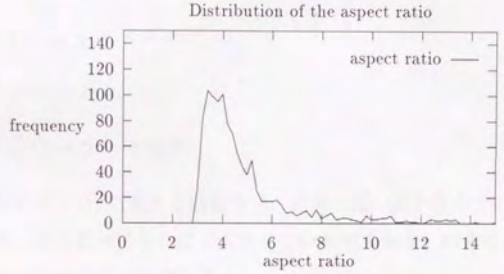


(c) 母点移動操作 2 回

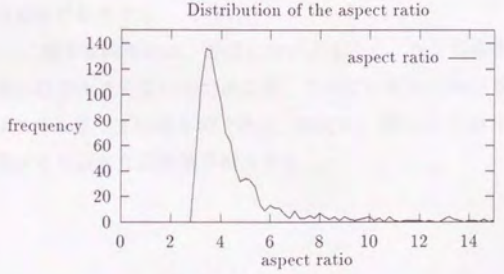
図 5.13. 面の内角の分布



(a) 初期母点配置に対する出力



(b) 母点移動操作 1 回



(c) 母点移動操作 2 回

図 5.14. 外接球と内接球の半径比の分布

5.5 考察

5.5.1 不適切な形状の四面体

ひとつの四面体において，それに外接する球の半径を R，最も長い辺を L，最も短い辺を l として，次の二つの値を定義する．

$$\omega = \frac{R}{L}$$

$$\kappa = \frac{L}{l}$$

これらの値と定数 α を使って，不適切な形状の四面体を次の三つのカテゴリに分類する [Dey, *et al.*, 1992]．

- カテゴリ (i) : $\omega \leq \alpha, \kappa > \alpha$.
- カテゴリ (ii) : $\omega > \alpha$.
- カテゴリ (iii): $\omega \leq \alpha, \kappa \leq \alpha$

すなわち，カテゴリ (i) に属する四面体は，非常に短い辺を含んでいて，かつ外接球の半径が一番長い辺と比べてそれほど大きくないものである．例えば，図 5.15 に示すような非常に尖った四面体が相当する．

カテゴリ (ii) に属する四面体は，外接球の半径が一番長い辺と比べて非常に大きいものである．例えば，図 5.16 に示すようなひとつの頂点の立体角が非常に大きく平たくなっている四面体が相当する．

カテゴリ (iii) に属する四面体は，非常に短い辺もなく，かつ外接球の半径が一番長い辺と比べてそれほど大きくないのであるが，すべての面の内角が 0 度（または 180 度）に近くつぶれてしまっているものである．例えば，図 5.17 に示すような外接円の中央近くに位置するつぶれた四面体が相当する．

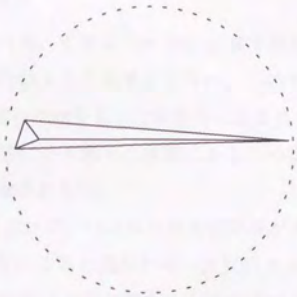


図 5.15. カテゴリー (i) に属する四面体

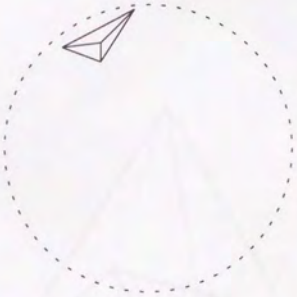


図 5.16. カテゴリー (ii) に属する四面体

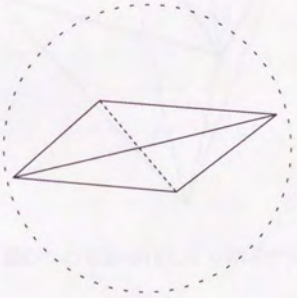


図 5.17. カテゴリー (iii) に属する四面体

5.5.2 対処できない場合

上の三つのカテゴリーのうち，カテゴリー (iii) に属する四面体に対しては，重心への移動を使った形状改善操作はあまり効果がでない．このカテゴリーに属する四面体は，図5.18に示すように適切な形状をもった四面体に囲まれて存在することがある．その場合には隣接母点とは十分につりあった状態にあるため重心への移動操作ではほとんど変化がなく，形状は改善されない．

5.4節で示した実験結果において，つぶれた四面体領域がある程度以下には減らないのは，この種の四面体の存在によると思われる．良質のメッシュを生成するためには，このような四面体に対する対策は不可欠であり，例えばつぶれた四面体の重心に新しい点を追加するなどの方策も考えられるが，詳しい解析は今後の課題として残されている．

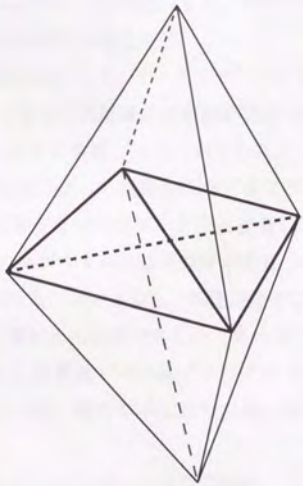


図 5.18. 重心への移動操作による効果が薄い場合

終章

Voronoi 図は広い応用範囲をもつ有益な幾何図形であり，高速な構成アルゴリズムも知られている．にもかかわらず，実用的なシステムに組み込まれ様々な応用分野で利用されているとはいい難い．というのも，理論的には正しいアルゴリズムであっても，それを計算機プログラムとして誤差のある世界で実行すると，処理の破綻の恐れがあるからである．

本論文では，3次元 Voronoi 図の構成法を対象として，位相優先法により数値的安定化を計ったアルゴリズムについて述べた．また，そのアルゴリズムを計算機へ実装し，計算機実験によりその有効性を確認した．

計算機プログラムの作成においては，テスト・デバッグ作業が予想以上に難航した．それは，2次元に比べ，3次元の位相構造は直観的把握が難しく，処理途中における位相構造の更新状況のチェックに手間どったためである．

しかしそれでも位相優先法に従って設計されたアルゴリズムは，退化した入力に対する例外処理を考慮する必要がないのでプログラムは著しく簡潔になっている．また，位相構造の更新を行なうアルゴリズムの骨格部分が完成すれば，それ以外の部分のデバッグ作業は格段に容易になる．というのも，処理が途中で異常終了したり，無限ループに陥ることがないため，常に出力結果が得られ，それをチェックすることができるからである．したがって，もし従来通りの方法でプログラミングしたとすれば，その労力はさらに大きくなるばかりか，動作が不安定で実用に耐えられないものしかできなかったであろう．

こうして実装されたプログラムを使った実験の結果，アルゴリズムの数値的安定性が実証されただけではなく，第3章で述べたように3次元の Delaunay 図と Voronoi 図における誤差の影響の差を新たに認識することができた．

さらに，第4章で述べた位相優先法を使った制約つき Delaunay 図の構成アルゴリズムも，計算機への実装や実験の過程においてそのアイデアを思いついた．グラフィッ

クスや有限要素法などの Delaunay 図の応用例を考えると，分割で使う面があらかじめ指定されるケースは多く，通常の Delaunay 図構成法からわずかな修正のみで制約つき Delaunay 図を近似的に構成できることによるメリットは大きい．

第5章ではメッシュ生成への応用例を紹介したが，ここでもシステムの数値的安定性が役立っている．有限要素法を使っている現場でのメッシュ生成は，一般に非常に多くの母点を与えるので計算時間がかかる．その際，処理の途中で数値誤差により処理のが破綻が生じると，入力データを修正してもう一度最初からやり直しになってしまう．ところが本アルゴリズムを利用すれば，とにかく解を出力してくれるので，オペレータはその結果をみてチェックできる．たとえ思い通りになっていない部分があっても対応がとりやすく作業効率が上がるはずである．

また，有限要素法における解の精度を上げるためには，各メッシュの形状が重要になってくる．ここでは，求まった Voronoi 図の位相構造を基に，最初に与えた母点を隣接する母点の重心に移すという操作を行ない改善を計った．しかし，全体的にはかなり良くなるものの，平たい四面体が残ってしまい，実用的なシステムを想定するとこれでは不十分であろう．実際の現場で話を聞くと，要素分割作業は，蓄積された多くのノウハウに基づいて行なわれているようであり，満足のいく自動要素分割を可能にするためにはクリアーしなければならない問題が多い．

謝辞

本研究の機会を与えてくださり，終始親切なご指導をいただきました東京大学大学院工学系研究科計数工学専攻の杉原厚吉教授に心から感謝いたします．

論文の審査にあたり，東京大学大学院工学系研究科計数工学専攻の伏見正則教授，速水　謙助教授，松井知己講師ならびに同精密機械専攻の木村文彦教授には，多忙な中，貴重な時間を割いて，多くのご指摘，ご助言をいただきました．ここに記して感謝の意を表します．

また，研究の便宜を計っていただき，いろいろと相談にのっていただいた名城大学の杉江　昇教授に厚く御礼申し上げます．

文部省内地研究員期間中は，東京大学大学院工学系研究科計数工学専攻の今井敏行助手，金子敬一助手から有益な助言を多数いただきました．ここに感謝の意を表します．

さらに，豊田工業高等専門学校の鬼頭幸生校長，堀井憲爾前校長，情報工学科の野沢繁之教授，仲野　貢教授をはじめとする同学科教職員各位，および大同工業大学の高田和之教授には，研究面でいろいろと便宜を計っていただくとともに，いつも激励の言葉をかけていただきました．謹んで感謝の意を表します．

また，堀情報科学振興財団ならびに内藤科学技術振興財団には，研究費の一部を助成していただきました．ここに記し感謝の意を表する次第であります．

最後に，研究生活を支えてくれた妻の幸子に感謝いたします．

参考文献

秋山　豊, 麻多　進, 熊代成孝, 田辺記生, 1993: 四面体 Delaunay 分割を用いた 3 次元配線シミュレータの開発. 電子情報通信学会技術研究報告, ED93-80 (SDM93-94, VLD93-35).

Aurenhammer, F., 1991: Voronoi diagram — A survey of a fundamental geometric data structure. *ACM Computing Surveys*, Vol. 23, pp. 345–401.

Benouamer, M., Michelucci, D., and Peroche, B., 1993: Error-free boundary evaluation using lazy rational arithmetic — A detailed implementation. *Proceedings of the 2nd Symposium on Solid Modeling and Applications*, Montreal, pp. 115–126.

Bern, M., and Eppstein, D., 1992: Mesh generation and optimal triangulation. In Du D.–Z., and Hwang F. K. (eds.) : *Computing in Euclidean Geometry*, pp. 23–90, World Scientific Publishing Co.

Brown, K. Q., 1979: Voronoi diagrams from convex hulls. *Information Processing Letters*, Vol. 9, pp. 223–228.

Cavendish, J. C., Field, D. A., and Frey, W. H., 1985: An approach to automatic three-dimensional finite element mesh generation. *International Journal for Numerical Methods in Engineering*, Vol., 21, pp. 329–347.

Dey, T., Bajaj, C., and Sugihara, K., 1992: On good triangulations in three dimensions. *International Journal of Computational Geometry and Applications*, Vol. 2, pp. 75–95.

Dobkin, D., and Silver, D., 1988: Recipes for geometric and numerical analysis — Part I, An empirical study. *Proceedings of the 4th Annual Symposium on Computational Geometry*, Urbana-Champaign, pp. 93–105.

Edelsbrunner, H., 1987: *Algorithms in Combinatorial Geometry*. Springer-Verlag, Berlin.

Edelsbrunner, H., and Seidel, R., 1986: Voronoi diagrams and arrangements. *Discrete and Computational Geometry*, Vol. 1, pp. 25–44.

Fortune, S., 1987: A sweepline algorithm for Voronoi diagrams. *Algorithmica*, Vol. 2, pp. 153–174.

Fortune, S., 1989: Stable maintenance of point-set triangulations in two dimensions. *Proceedings of the 3-th IEEE Annual Symposium on Foundations of Computer Science*, Research Triangle Park, California, pp. 494–499.

藤本忠博, 大野義夫, 1992: 不規則形状生成のためのモデリング手法. 情報処理学会グラフィックスと CAD 研究会研究報告, CG-56-5, pp. 33–40.

Greene, D. H, and Yao, F., 1986: Finite-resolution computational geometry. *Proceedings of the 27th IEEE Annual Symposium on Foundations of Computer Science*, Toronto, pp. 143–152.

Guibas, L., Salesin, D., and Stolfi, J., 1989: Epsilon geometry — Building robust algorithms from imprecise calculations. *Proceedings of the 5th ACM Annual Symposium on Computational Geometry*, Saarbrücken, pp. 208–217.

Guibas, L., and Stolfi, J., 1985: Primitives for the manipulation of general subdivisions and the computation of Voronoi diagrams. *ACM Transactions on Graphics*, Vol. 4, pp. 74–123.

Hoffman, C., Hopcroft, J., and Karasick, M., 1988: Towards implementing robust geometric computations. *Proceedings of the 4th ACM Annual Conference on Computational Geometry*, Urbana-Champaign, pp. 106–117.

Hoffman, C., 1989: The Problems of accuracy and robustness in geometric computation. *Computer*, Vol. 22, No. 3, pp. 31–41.

Ho-le, K., 1988: Finite element mesh generation methods: a review and classification. *computer-aided design*, Vol. 20, No. 1, pp. 27–38.

今井敏行, 杉原厚吉, 1994: 誤差による破綻の心配のない線分 Voronoi 図構成算法. 情報処理学会論文誌, Vol. 35, No. 10, pp. 1966–1977.

Inagaki, H., Sugihara, K., and Sugie, N., 1992: Numerically robust incremental algorithm for constructing three-dimensional Voronoi diagrams. *Proceedings of the 4th Canadian Conference on Computational Geometry*, St. John's, pp. 334–339.

稲垣　宏, 杉原厚吉, 杉江　昇, 1994: 3 次元ボロノイ図構成のための数値的に安定な逐次添加法. 情報処理学会論文誌, Vol. 35, No. 1, pp. 1–10.

稲垣　宏, 杉原厚吉, 1994a: 退化を許す3次元 Delaunay 図構成算法. 情報処理学会グラフィックスと CAD 研究会研究報告, Vol. 94, No. 17, CG-67, pp. 9–16.

稲垣　宏, 杉原厚吉, 1994b: 制約つき Delaunay 図構成算法. 情報処理学会アルゴリズム研究会研究報告, Vol. 94, No. 26, AL-38, pp. 41–48.

Inagaki, H., and Sugihara, K., 1994: Numerically robust algorithm for constructing constrained Delaunay triangulation. *Proceedings of the 6th Canadian Conference on Computational Geometry*, Saskatoon, pp. 171–176.

伊理正夫 (監), 腰塚武志 (編), 他, 1993: 計算幾何学と地図情報処理. 第2版, 共立出版, 東京.

伊理正夫, 杉原厚吉, 1988: 計算誤差を考慮した幾何的アルゴリズム. 情報処理学会アルゴリズム研究会研究報告, AL-1-1.

Karasick, M., Lieber, D., and Nackman, L. R., 1991: Efficient Delaunay triangulation using rational arithmetic. *ACM Transactions on Graphics*, Vol. 10, pp. 71–91.

Lee. D. T., and Schachter, B. J., 1980: Two algorithms for constructing a Delaunay triangulation. *International Journal of Computer and Information Sciences*, Vol. 9, pp. 219–242.

Milenkovic, V., 1988: Verifiable implementations of geometric algorithms using finite precision arithmetic. *Artificial Intelligence*, Vol. 37, pp. 377–401.

Milenkovic, V., 1989: Double precision geometry — A general technique for calculating line and segment intersections using rounded arithmetic. *Proceedings of the 30th IEEE Annual Symposium on Foundations of Computer Science*, pp. 500–505

名取　亮, 野寺　隆 (編), 1989: 小特集「グリッド・ジェネレーションとその応用」. 情報処理, Vol. 30, pp. 766–795.

大石泰章, 杉原厚吉, 1991: 数値的に安定な分割統治型ボロノイ図構成算法. 情報処理学会論文誌, Vol. 32, No. 6, pp. 709–720.

Ohya, T., Iri, M., and Murota, K., 1984: Improvements of the incremental method for the Voronoi diagram with computational comparison of various algorithms. *Journal of the Operations Research Society of Japan*, Vol. 27, pp. 306–336.

Okabe, A., Boots, B., and Sugihara, K., 1992: *Spatial Tessellations —Concepts and Applications of Voronoi Diagrams*. John Wiley & Sons, Chichester.

Ottmann, T., Thiemt, G., and Ullrich, C., 1987: Numerical stability of geometric algorithms. *Proceedings of the 3rd ACM Annual Symposium on Computational Geometry*, Waterloo, pp. 119–125.

Preparata, F. P., and Shamos, M. I., 1985: *Computational Geometry — An Introduction*. Springer-Verlag, NewYork.

Schumaker, L., 1993: Triangulations in CAGD. *IEEE Computer Graphics and Applications*, Vol. 13, No. 1, pp. 47–52.

Segal, M., and Sequin, C. H., 1985: Consistent calculations for solid modeling. *Proceedings of the ACM Symposium on Computational Geometry*, Baltimore, pp. 29–38.

Shamos, M. I., and Hoey, D., 1975: Closest-point problems. *Proceedings of the 16th IEEE Annual Symposium on Foundations of Computer Science*, pp. 151–162.

杉原厚吉, 1991: 幾何アルゴリズムの数値的破綻とその対策. 応用数理, Vol. 1, No. 4, pp. 280–299.

杉原厚吉, 1994: 計算幾何工学. 培風館, 東京.

杉原厚吉, 伊理正夫, 1987: 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.

Sugihara, K., 1992: A simple method for avoiding numerical errors and degeneracy in Voronoi diagram construction. *IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences*, Vol. E75-A, pp. 468–477.

Sugihara, K., 1994a: A robust and Consistent Algorithm for intersecting Convex polyhedra. *EUROGRAPHICS '94*, Vol. 13, No. 3, pp. C45–C54.

Sugihara, K., 1994b: Robust gift wrapping for the three-dimensional convex hull. *Journal of Computer and System Sciences*, Vol. 49, No. 2, pp. 391–407.

Sugihara, K., and Inagaki, H., 1995: Why is the 3D Delaunay triangulation difficult to construct?. *Information Processing Letters*, Vol. 54, pp. 275–280.

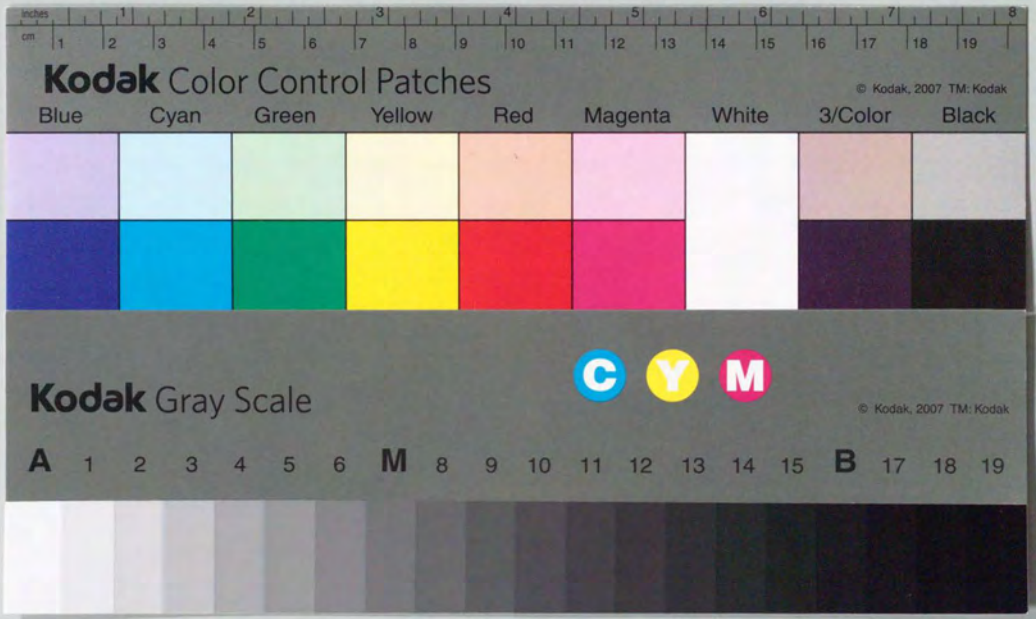
Sugihara, K., and Iri, M., 1992: Construction of the Voronoi diagram for "one million" generators in single-precision arithmetic. *Proceedings of the IEEE*, Vol. 80, pp. 1471–1484.

Sugihara, K., and Iri, M., 1994: A robust topology-oriented incremental algorithm for Voronoi diagrams. *International Journal of Computational Geometry & Applications*, Vol. 4, No. 2, pp. 179–228.

谷口健男, 太田　親, 1991: 三次元凸体の四面体有限要素自動分割. 土木学会論文集, No. 432/I-16, pp. 137-144.

山本裕之, 内山晋二, 田村秀行, 1995: 3次元形状モデリングのためのドロネー網生成法. 電子情報通信学会論文誌, Vol. J78-D-II, No. 5, pp. 745–753.

吉田清範, 1986: 代数的な量の符号判定に必要な計算精度. 電子通信学会論文誌, Vol. J69-A, pp. 543–547.


inches
cm
Kodak Color Control Patches
© Kodak, 2007 TM: Kodak
Blue
Cyan
Green
Yellow
Red
Magenta
White
3/Color
Black
Kodak Gray Scale
C
Y
M
© Kodak, 2007 TM: Kodak
A 1 2 3 4 5 6 M 8 9 10 11 12 13 14 15 B 17 18 19