

## 制約つき 3 次元 Delaunay 図の構成算法

稲垣 宏

豊田工業高等専門学校  
情報工学科

杉原 厚吉

東京大学工学部  
計数工学科

筆者らは、文献 [1][2] において、位相構造を優先することで数値的安定化を図った 3 次元 Voronoi 図構成法を提案した。また、文献 [3] では、このアルゴリズムを利用して、双対図形である 3 次元 Delaunay 図を生成する方法を提案した。本稿では、位相構造を優先させた Voronoi 図構成法が、制約つき Delaunay 図の構成にも利用できることを示す。ここで提案する算法では、制約を満たすように数値判定をねつ造しているために、正しいボロノイ図は出力されない。しかし、その双対を求めると、制約つき Delaunay 図になっているのである。また、本算法を適用した 3 次元の制約つき Delaunay 図構成プログラムを作成し、計算機実験により、その有効性を実証した。

### Three-Dimensional Constrained Delaunay Triangulation Algorithm

Hiroshi Inagaki<sup>†</sup> and Kokichi Sugihara<sup>††</sup>

<sup>†</sup>Department of Information and Computer Engineering  
Toyota National College of Technology  
2-1 Eisei-cho, Toyota-shi, Aichi 471 Japan

<sup>††</sup>Department of Mathematical Engineering and Information Physics  
Faculty of Engineering, University of Tokyo  
7-3-1 Hongo, Bunkyo-ku, Tokyo 113, Japan

In our previous paper[2], we proposed a numerically robust algorithm for constructing the three-dimensional Voronoi diagram. In this paper, we show that this algorithm can be applied not only to the construction of the ordinary Delaunay triangulation but also to the construction of the constrained Delaunay triangulation. In order to construct the constrained Delaunay triangulation, we invent numerical results on the way to the construction of the Voronoi diagram in such a way that they satisfy the constraints. Moreover we implemented the algorithm as a computer program, and verified the validity of the algorithm by computational experiments.

## 1 はじめに

Delaunay 図 (Delaunay triangulation) は, Voronoi 図 (Voronoi diagram) の双対図形であり, 入力された母点の凸包の三角形分割を与える. 制約つき Delaunay 図 (constrained Delaunay triangulation) とは, 指定されたいくつかの辺を必ず使わなければならないという制約のもとで定義される Delaunay 図である. そして, これらの図形は, 有限要素法のためのメッシュ生成, 補間, 経路計画など, 広い分野での応用をもっている [8][9].

そのため, Voronoi 図や Delaunay 図の構成に関して, いくつかの効率の良いアルゴリズムが提案されてきた. しかし, それらのアルゴリズムは, 計算誤差のない世界で考えられたものであるために, そのまま計算機プログラムに翻訳しても, 必ずしも正常に動作するとは限らない. 計算過程における誤差が処理の破綻を引き起こすことがあり, 実用的なシステムを構成できないのである.

このような問題は, Voronoi 図や Delaunay 図に限ったものではなく, 幾何図形を扱うアルゴリズム一般について生じてくる. 最近になって, このような理論と現実とのギャップを埋めようとする試みが, 具体的な幾何的アルゴリズムを対象に行なわれつつある [1][2][3][4][5][11][12].

本稿では, 筆者らが提案している位相構造を優先させた Voronoi 図構成アルゴリズムが, 制約つき Delaunay 図の構成にも利用できることを示す. さらに, ここで提案するアルゴリズムを 3 次元の制約つき Delaunay 図構成プログラムとしてインプリメントを行なったので, その計算機実験結果も合わせて報告する.

## 2 数値的に安定な 3 次元 Voronoi 図構築法

### 2.1 Voronoi 図と Delaunay 図

$n$  次元空間の有限点集合  $S = \{p_1, p_2, \dots, p_k\}$  に対して,  $V(p_i) = \{x \mid x \in R^n, d(x, p_i) < d(x, p_j), j = 1, 2, \dots, k, j \neq i\}$  (ただし,  $d(x, p_i)$  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 領域といい,  $n$  次元空間の Voronoi 領域  $V(p_1), V(p_2), \dots, V(p_k)$  への分割を,  $S$  に対する Voronoi 図という. このとき,  $S$  の要素を母点という. Voronoi 領域  $V(p_i)$  は,  $S$  に属す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  $p_i$  の勢力圏と呼ばれることもある [6][7].

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, これが Delaunay 図と呼ばれる図形である. 2 次元の場合の例を図 1 に示す. 黒丸で表した母点に対する Voronoi 図を破線で示し, Delaunay 図を実線で示してある.

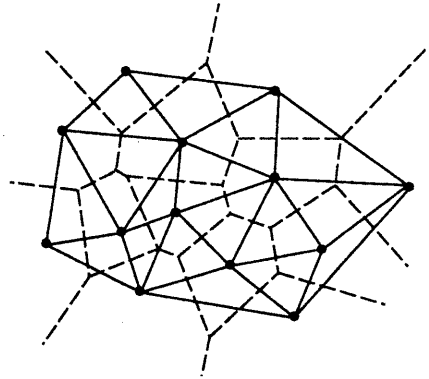


図 1. 2 次元の Voronoi 図と Delaunay 図

3 次元の場合には, Voronoi 領域は多面体となり, Voronoi 図は 3 次元空間の多面体分割を与える. また, Delaunay 図は, 母点集合の凸包を, 母点を頂点とする四面体に分割する. 本稿では, Voronoi 図における多面体領域を Voronoi 領域 (Voronoi region), その頂点を Voronoi 点 (Voronoi point) と呼び, Delaunay 図における四面体を Delaunay 四面体 (Delaunay tetrahedron) と呼ぶことにする.

### 2.2 Voronoi 図構成アルゴリズム

これまで提案されている Voronoi 図構成アルゴリズムのうちで, 実用的な意味で最も効率が良いとされているのが母点逐次添加法である. これは, 数個の母点に対する Voronoi 図から出発して, 母点を一つずつ追加しながら Voronoi 図を逐次更新していく方法である. この方法を使えば, データ構造上の工夫により, ランダムに配置された  $n$  個の母点に対しては, 平均的に  $O(n)$  の時間で Voronoi 図を構成できる [10].

### 2.3 数値的安定化

母点逐次添加法をはじめとする従来の Voronoi 図構成アルゴリズムをそのまま計算機にインプリメン

トしても、計算誤差が原因となって処理が破綻することがあり、実用的なシステムを構築できない。特に母点同士が特殊な位置関係（退化と呼ばれる）にあるとき、または、それに近いときには、その危険性が高くなる。

筆者らは、母点逐次添加法を基に数値的安定化を計った Voronoi 図構成アルゴリズムを提案した。そこでの基本方針は、母点添加時の Voronoi 図更新作業において、数値計算結果より Voronoi 図の有する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ごうとするものである。この方針を採用したアルゴリズムは、2次元 Voronoi 図に関しては文献 [4][5] に、3次元 Voronoi 図に関しては文献 [1][2] に詳しい。ここでは、3次元 Voronoi 図構築における中心的な処理である母点添加に伴う Voronoi 図更新のアルゴリズムについてその概略を示すにとどめる。例を図 2 に示し、各ステップに対するこの例での説明を括弧の中に記した。

#### 母点添加時の Voronoi 図更新アルゴリズム

1. 新しい母点の添加に伴い削除すべき Voronoi 点の集合を選ぶ。（図 2 (a) において、二重四角で表されている点々が、新たに添加された母点であり、黒丸が削除すべき点であるとして選ばれた Voronoi 点であるとする。ここでは、図を簡単にするため、削除すべき点であるとして選ばれた Voronoi 点がこの 1 点のみである場合について説明する。）
2. 削除すべき Voronoi 点とそれに隣接する残される Voronoi 点を結ぶ辺上に、新しい Voronoi 点を作る。（同図 (b) では、この新しい Voronoi 点を白丸で示してある。）
3. Voronoi 領域ごとに、新しく作られた Voronoi 点を結んで、新しい辺と面を作る（同図 (c)）。
4. 新しい面で囲まれた領域の内部を削除し、新しい多面体領域を作る（同図 (d)）。これを添加された母点に対する Voronoi 領域とみなし、更新作業を終了する。

ここで、ステップ 1 における Voronoi 点の選び方には任意性があるので、最も確からしいものを選ぶために数値計算を行なう。具体的には、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、新しく添加された母点が含まれるかどうかを計算する。もし、内部に含まれれば、その Voronoi 点は新しい母点の勢力圏に入るため削除される。ただし、

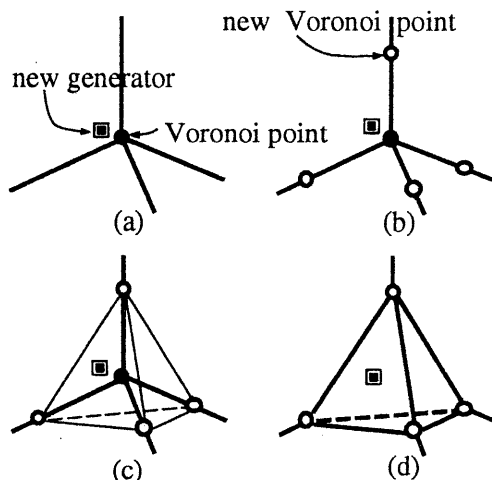


図 2. 母点添加時の Voronoi 図の更新

Voronoi 図の位相構造に矛盾をきたすような数値判定がなされた場合には、その判定には従わないことにする。このように、位相構造の無矛盾性を優先させて処理を進めることで、計算誤差による処理の破綻を防いでいるのである。

### 3 制約つき Delaunay 図の構成

本稿では、3次元の制約つき Delaunay 図を構築することを目的としているが、説明が煩雑になるのを避けるために、ここでは、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ことにする。

#### 3.1 有界 Voronoi 図

$S$  を  $R^2$  における  $n$  個の母点の集合とし、 $O$  を  $S$  の点を端点とする  $m$  本の互いに交差しない線分の集合とする。 $O$  に属す線分を障害物とみなし、2点  $x, y \in R^2$  の有界距離 (bounded distance) を

$$b(x, y) = \begin{cases} d(x, y) & \overline{xy} \cap O = \emptyset \text{ のとき,} \\ \infty & \text{それ以外の場合} \end{cases}$$

と定義する。ただし、 $d(x, y)$  は、点  $x$  と点  $y$  のユークリッド距離を表し、 $\overline{xy}$  は、 $x$  と  $y$  を両端とする閉線分を表す。

ここで、すべての  $p \in S$  に対して、

$$V_b(p) = \{x \in R^2 \mid b(x, p) \leq b(x, q), q \in S\}$$

で定義される領域によって、 $S$  と  $O$  に関する有界 Voronoi 図が与えられる [8]. 有界 Voronoi 図においては、 $O$  に属す線分の端点となっている母点の Voronoi 領域は一般には凸にならないのに対して、それ以外の母点の Voronoi 領域は凸になっている。また、有界ポロノイ図の双対図形は、与えられた母点の凸包内部の三角形分割を与えるとは限らない。図 3 に有界 Voronoi 図およびその双対図形の例を示す。障害物となる線分を太実線で、有界ポロノイ図を破線で、その双対図形を細実線で表している。灰色で塗られているのは、三角形分割がなされていない領域である。

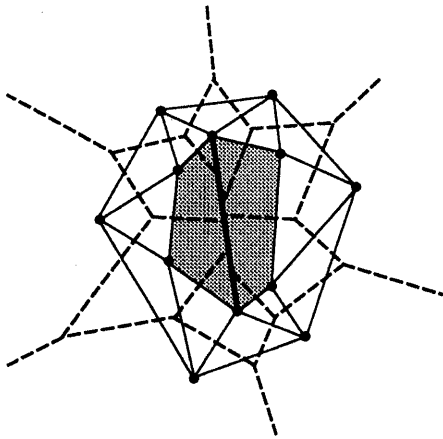


図 3. 有界 Voronoi 図

### 3.2 制約つき Voronoi 図と制約つき Delaunay 図

有界 Voronoi 図における Voronoi 領域のうち、各障害物  $o \in O$  を辺としてもつような領域を拡張することによって、制約つき Voronoi 図へ変更することができる [8]. 障害物の左に隣接している領域の母点を  $p_1, p_2, \dots, p_k$  とし、障害物の右に隣接している領域の母点を  $q_1, q_2, \dots, q_l$  とする。ただし、障害物となっている線分の端点は、両側に隣接しているものと考え、両方の集合に属している。ここで、左に隣接している領域に対しては、母点  $p_1, p_2, \dots, p_k$  のみがあるものとして、障害物の右側へ領域を拡張する。また、障害物の右側に隣接している領域に対しては、母点  $q_1, q_2, \dots, q_l$  のみがあるものとして、障害物の左側へ領域を拡張する。したがって、各障

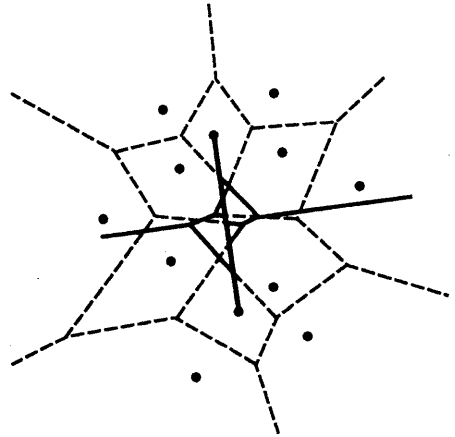


図 4. 制約つき Voronoi 図

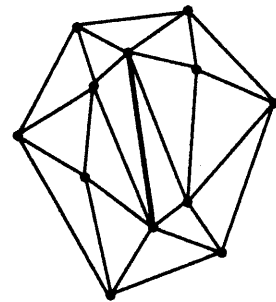


図 5. 制約つき Delaunay 図

害物の端点となっている母点の領域は、両側へ拡張されることになる。

例えば、図 3 の有界 Voronoi 図から変形された制約つき Voronoi 図を図 4 に示す。障害物となる線分および拡張された領域を太実線で示してある。このように、制約つき Voronoi 図の Voronoi 領域は、一般には重なり合う部分が生じ、平面グラフ構造にはならない。そのため、制約のない場合に比べて位相構造が複雑になり、数値的に安定なアルゴリズムを構築することがむずかしそうに思える。

しかし、制約つき Voronoi 図の有効性は、その図そのものよりも、双対図形である制約つき Delaunay 図の生成にある。図 4 に対する制約つき Delaunay 図を図 5 に示す。次章では、位相構造を優先させたアルゴリズムの特徴をうまく利用すれば、制約つき Voronoi 図を作らなくても、普通の Voronoi 図の位相構造のみを使って、制約つき Delaunay 図を生成できることを示す。

### 3.3 位相構造を優先させた Voronoi 図の利用

2.3 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 6 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（この母点配置および制約条件は図 3 ~ 5 と同じである）。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、これまでどお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 7 に示す。数値判定をねつ造したために中央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 5 に示した制約つき Delaunay 図が得られる。

このような手法が可能であるのは、位相構造を優先する我々の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ため、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。今回提案する手法では、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。

### 3.4 3次元への拡張

3次元の制約つき Delaunay 図は、指定された三角面を必ず使った凸包内部の四面体分割を与える。前節で述べた基本方針は、3次元の場合にもそのまま適用することができる。この基本方針に従ったアル

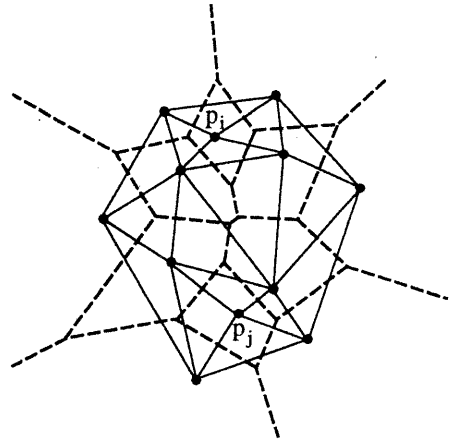


図 6. 制約なしの Voronoi 図と Delaunay 図

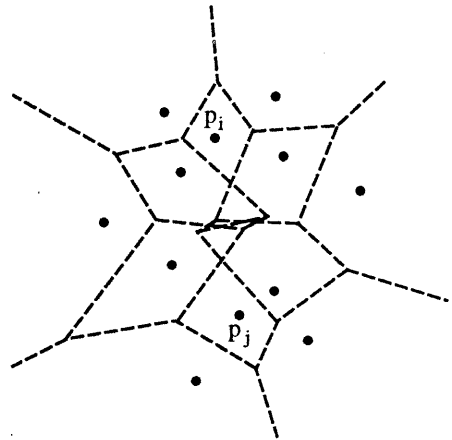


図 7. ねじれた Voronoi 図

ゴリズムの出力の様子については、次節において計算機実験結果を使って見ていくことにする。

なお、3次元の場合には、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が、そのような場合の対処については検討中である。

## 4 計算機実験

3節で述べた手法を3次元に拡張したものを、2節で紹介した3次元 Voronoi 図構成アルゴリズムに適用し、3次元の制約つき Delaunay 図構成プログラムを作成した。本節では、計算機実験結果をもとに、そのプログラムの振舞いをみていく。

#### 4.1 簡単な例による動作確認

まず、図 8 に示すような 6 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 9 に示す。そして、この結果をもとに生成された Delaunay 図を、図 10 に示す。なお、出力結果は、両眼立体視（視線交差法）用の画像として表示する（以下の図も同様である）。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 11 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 12）。この結

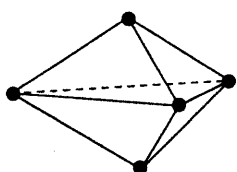


図 8. 母点配置（制約なし）

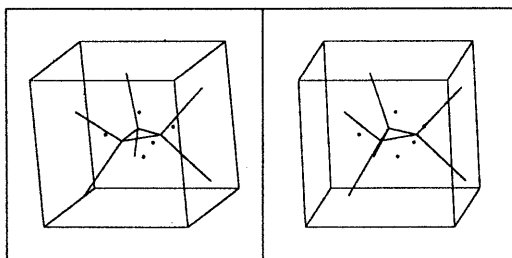


図 9. 制約をつけなかったときの Voronoi 図

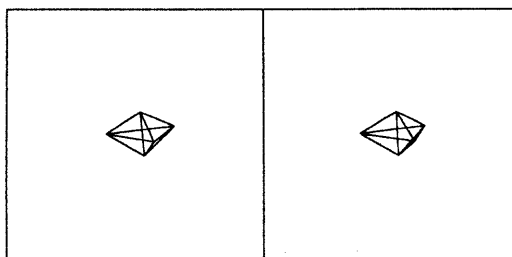


図 10. 制約をつけなかったときの Delaunay 図

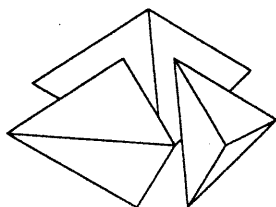


図 11. 制約をつけなかったときの分割のようす

果得られた Voronoi 図を図 13 に示す。これを見ると、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 14 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 15 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。

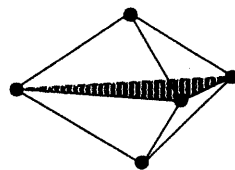


図 12. 制約となる面

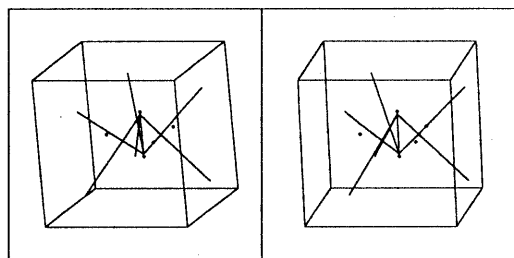


図 13. 制約をつけたときの Voronoi 図

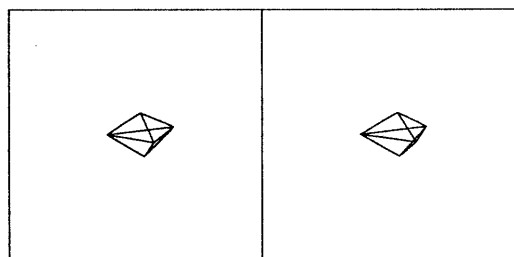


図 14. 制約をつけたときの Delaunay 図

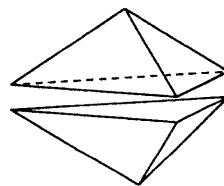


図 15. 制約をつけたときの分割のようす

#### 4.2 多面体を制約とした場合

まず、図 16 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

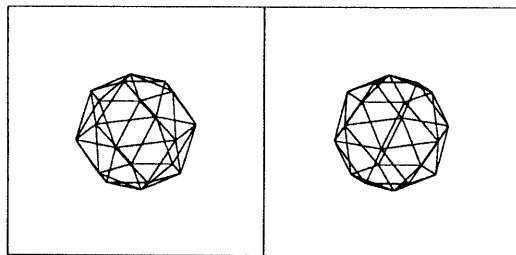


図 16. 制約となる多面体

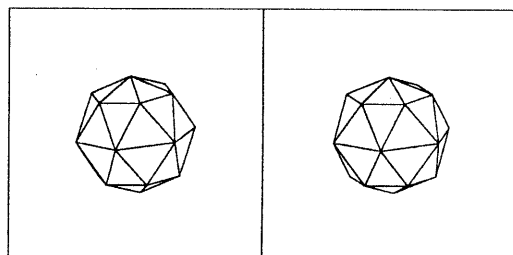


図 20. 残った多面体

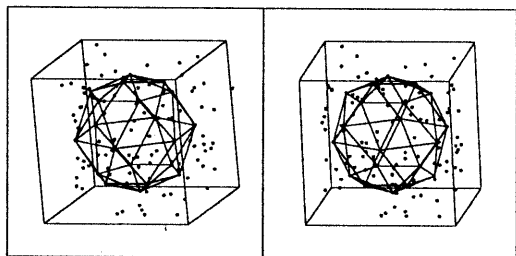


図 17. 入力データ

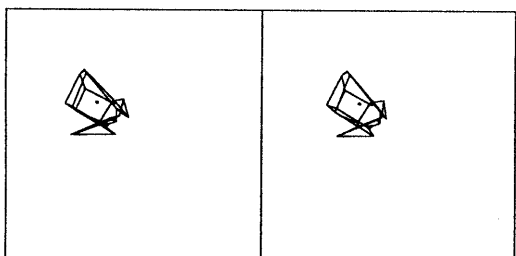


図 18. 反転部分を含む Voronoi 領域

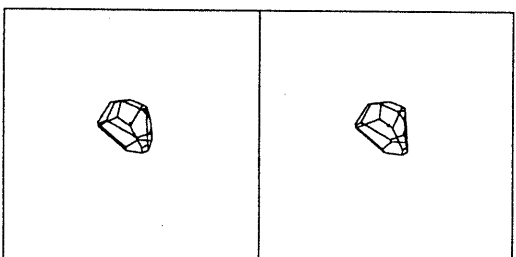


図 19. 反転部分のない Voronoi 領域

の多面体の内外にランダムに母点を発生させる。図 17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 18, 図 19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 18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 19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 図を生成すると、多面体の面をすべて使った四面体分割になっているはずである。そこで、得られた各四面体の重心が、多面体の外にあるものをすべて取り去ったものが、図 20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る。

### 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 21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 22)。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 23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。

## 5 おわりに

本稿では、位相構造を優先させた Voronoi 図構成アルゴリズムを利用すると、制約のない Voronoi 図の位相構造をそのまま用いて、制約つきの Delaunay 図を生成することができることを示した。そこでの

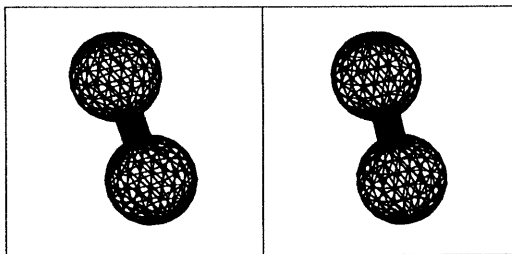


図 21. 制約となる多面体

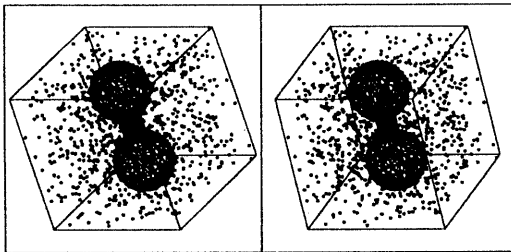


図 22. 入力データ

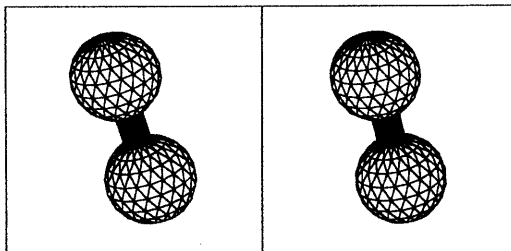


図 23. 残った多面体

基本方針は、位相構造優先法の特徴である“どんなに大きな計算誤差が含まれていても位相的には矛盾のない図形が出力される”ことを積極的に利用し、与えられた制約を満たすように数値判定値をねつ造してしまふことである。この工夫により、制約なしの Voronoi 図構成プログラムの数値判定部を修正するだけで、制約つき Delaunay 図を数値的に安定に生成することができる。

実際に、3次元 Voronoi 図構成プログラムに本工夫をインプリメントし、計算機実験を行なった結果、本アルゴリズムの有効性を確認することができた。よって、本アルゴリズムが、要素分割、補間、経路計画、グラフィックスなどの問題に対して、強力なツールを提供することができると期待される。

謝辞 本研究の一部は文部省科学研究費補助金 (No. 05780271) と内藤科学技術研究助成金の援助による。

## 参考文献

- [1] 稲垣 宏, 杉原厚吉, 杉江 昇: “3次元ポロノイ図構成のための数値的に安定な逐次算法”, 情報処理学会論文誌, **35**, 1, pp. 1 - 10 (1994).
- [2] Inagaki, H., Sugihara, K. and Sugie, N.: “Numerically Robust Incremental Algorithm for Constructing Three-Dimensional Voronoi Diagrams”, Proc. Fourth Canadian Conference on Computational Geometry, pp. 334 - 339 (1992).
- [3] 稲垣 宏, 杉原厚吉: “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研究報告, 94 - CG - 67, pp. 9 - 16 (1994).
- [4] 杉原厚吉: “幾何アルゴリズムの数値的破綻とその対策”, 応用数理, **1**, 4, pp. 280-299 (1991).
- [5] Sugihara, K. and Iri, M.: “Construction of the Voronoi Diagram for One Million Generators in Single-Precision Arithmetic”, Proc. IEEE, **80**, pp. 1471 - 1484 (1992).
- [6] Preparata, F. P. and Shamos, M. I.: “Computational Geometry — An Introduction”, Springer Verlag, New York (1985).
- [7] 伊理正夫 (監), 腰塚武志 (編), 他: “計算幾何学と地図情報処理 第2版”, 共立出版, 東京 (1993).
- [8] Aurenhammer, F.: “Voronoi Diagrams — A Survey of a Fundamental Geometric Data Structure”, ACM Computing Survey, **23**, 3, pp. 345 - 405 (1991).
- [9] Bern, M. and Eppstein, D.: “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. (1992).
- [10] Ohya, T., Iri, M. and Murota, K.: “Improvements of the Incremental Method for the Voronoi Diagram with Computational Comparison of Various Algorithms”, J. Operation Research Society of Japan, **27**, 4, pp. 306 - 336 (1984).
- [11] Milenkovic, V.: “Verifiable Implementations of Geometric Algorithms Using Finite Precision Arithmetic”, Artificial Intelligence, **37**, pp. 377 - 401 (1988).
- [12] Hoffman, C., Hopcroft, J. and Karasick, M.: “Towards Implementing Robust Geometric Computations”, Proc. 4th ACM Annual Conference on Computational Geometry, pp. 106 - 117 (1988).