計算幾何学と建築

Computational Geometry and Architecture

加藤 直樹

Naoki Katoh

1. はじめに

計算幾何学は、主として2次元平面や3次元空間における幾何学的データ（点、直線、線分、円、平面、球など）を対象としたデータ処理に関連するさまざまな問題を計算機を用いて高速かつ効率的に解くためのアルゴリズムの開発や、その限界を究明する研究分野である。1975年頃から、理論の基礎的体系が整備されて以来、コンピュータグラフィックス，CAD，地理情報処理，ロボティックス，パターン認識などの関連応用分野の進展とあいまって、急速な発展を遂げた。計算幾何学は、計算機科学の一分野であるが、過度に抽象的な対象を相手にするのではなく、(初等) 幾何学で現れる図形を対象としているため、具体性がありなじみやすいということもあって、たちまち流行し発展した。日本でも多くの研究者がこの分野に取り組み、日本の計算幾何学の研究レベルは高く、良書もいくつか出ている [1, 3–5, 7, 8, 11, 12]。Traverse の読者の多くは、「計算幾何学」という言葉を初めて耳にしたことと思うので、本稿ではまずはじめに計算幾何学の幾つかの基本的な話題について説明し、その後建築への応用について筆者が最近おこなった研究について解説する。

「計算幾何学はどんなことをするの?」と思う方も多いと思う。たとえば、都市工学では小売店の商圏分析、公立小学校の校区の妥当性の検討などで、ボロノイ図をよく用いる。小売店の商圏分析の場合、平面上の任意の点に対する最寄の店はどこかということを表す勢力圏図のことである (図4参照)。このようなボロノイ図を計算したり、点集合の凸包や三角形分割を求めたりすることは計算幾何学の基本の一つである (図5は、三角形分割の例を示す)。また、近年では、地図情報を計算機に蓄積し、さまざまな検索をおこなう地理情報システムの開発がおこなわれており、手頃な価格で購入できる。たとえば、ある地点から半径 1km 以内にあるコンビニエンスストアを列挙するというようなことも手軽にできる。このような検索の高速計算を実現するためには、計算幾何学の分野で開発された幾何学的データ構造が用いられている。一般的にはアルゴリズム理論は、与えられた問題に対して、その問題を効率よく解く方法（手順とかアルゴリズム）を構築したり、そのための一般的な方法論を研究することを目的としている。また、問題によっては効率よく解けないものもあり、これ以上は速く解く方法はないという本質的な問題の難しさを研究する学問でもある。しかし、計算幾何学で扱う問題は素人でも理解しやすく、多くの基本的問題に対しては、その問題を解く方法を思いつくのは、プログラムを作った経験のある人には難しいことではない。それにもかかわらず、計算幾何学の分野で開発されたアルゴリズムやデータ構造は素人では思いつかないさまざまな着想、理論的な深さがあり、人々を魅了させるものがある。

建築分野において、計算幾何学は、ヴァーチャルリアリティにおける建築空間のウォークスルー、コンピュータグラフィックスにおけるレンダリング、コンピュータビジョンにおける、地理情報システム (GIS) におけるさまざまなオブジェクトの高速検索などに応用されているが、基礎技術のなかに用いられていることが多いので、一般の人には建築と計算幾何学の関係が表立って見えることはない。本稿では、まずはじめに計算幾何学が取り扱う基礎的な問題の一部を紹介し、その後、筆者らが最近取り組んだ、三角形トラスの設計の応用に関連する一様三角形メッシュ生成問題とその研究成果について紹介する。

2. 基本的問題

2.1 凸包

与えられた d 次元空間の n 点集合 S に対して、S のすべての点を含む最小の凸多面体を凸包 (convex hull) という。計算幾何学では最も基本的な問題で、深く研究されている。ここでは 2 次元平面の場合について簡単に述べる。図 1 の点集合 S に対して、S の凸包は図 2 のようになる。

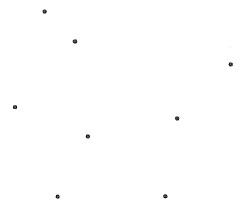

図 1　点集合 S

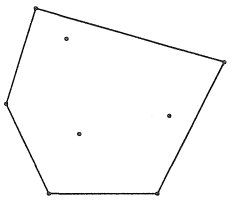

図 2　S の凸包

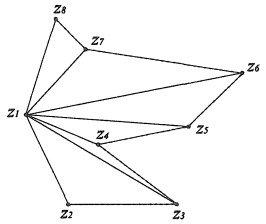

図 3: Graham のアルゴリズムの説明図

2 次元凸包を計算するアルゴリズムは多数提案されているが、ここでは理解しやすく、速い Graham の方法 [6] を紹介する。まず凸包の内部の点を原点とし（ここでは x 座標が最小の S の点を選ぶことにする）、それを z_1 として、その点を中心に偏角が小さい順に $z_2, z_3, \ldots, z_n$ として、内角 $z_i z_{i+1} z_{i+2}$ が 180° 未満かどうかを調べながら凸包を構成するものである。アルゴリズムでは、点のリスト L を保持しながら進む。最初は $L = z_1 z_2 z_3$ である。いつでもリストの最後にある 3 点 (u, v, w とする) に対して内角 $\alpha = uvw$ を調べ、α が 180° 未満なら $z_2, z_3, \ldots, z_n$ のなかでまだ調べていない最も添え字の番号の小さい z_i をリストの最後尾に追加する。180° 以上なら v を削除し残されたリストに対して、内角が 180° 未満となるまで、同じ手続きを繰り返す。具体的に図 3 を用いて説明しよう。はじめに、$z_1 z_2 z_3$, $z_2 z_3 z_4$ のいずれの内角も 180° 未満であるが、$z_3 z_4 z_5$ は 180° 以上である。このとき、z_4 は凸包上にはないと判断され、これを削除し、$z_2 z_3 z_5$ の内角を調べる。これは 180° 未満であるので次に $z_3 z_5 z_6$ の内角を調べる。以下同様の手続きが繰り返される。点のリストの変化の状況を次の表 1 で示す。最後に残っている点のリストが凸包上の反時計回りの点列である。

表 1　Graham のアルゴリズムを図 3 にしたがって実行したときのリストの変化の状況

追加した点	調べている内角	リスト
z_1		z_1
z_2		$z_1 z_2$
z_3	$z_1 z_2 z_3$	$z_1 z_2 z_3$
z_4	$z_2 z_3 z_4$	$z_1 z_2 z_3 z_4$
z_5	$z_3 z_4 z_5$	$z_1 z_2 z_3 z_4 z_5$
	$z_2 z_3 z_5$	$z_1 z_2 z_3 z_5$
z_6	$z_3 z_5 z_6$	$z_1 z_2 z_3 z_5 z_6$
	$z_2 z_3 z_6$	$z_1 z_2 z_3 z_6$
z_7	$z_3 z_6 z_7$	$z_1 z_2 z_3 z_6 z_7$
z_8	$z_6 z_7 z_8$	$z_1 z_2 z_3 z_6 z_7 z_8$
	$z_3 z_6 z_8$	$z_1 z_2 z_3 z_6 z_8$
z_1	$z_6 z_8 z_1$	

このアルゴリズムの計算時間は $O(n \log n)$ である。この意味は、基本的演算（加減乗除、比較、代入の演算）の回数が点の数の $n \log n$ に比例するということである。最初に偏角順

に点を並べる必要があるが、この部分が最も時間の掛かるところであり、$O(n \log n)$ 時間を要する。調べる三角形の数は全体の点の数の高々2倍なので、残りの部分の計算時間は $O(n)$ である。

2.2 ボロノイ図

平面上の n 点 $P_i(x_i, y_i), i = 1, 2, \ldots, n$ が与えられたとき、点 P_i の勢力圏 $V_n(P_i)$ を

$$V_n(P_i) = \cap_{j \neq i}\{P \mid d(P, P_i) < d(P, P_j)\} \tag{1}$$

($d(P, P_i)$ は点 P と P_i のユークリッド距離である) で定義し、これを点 P_i に対するボロノイ多角形という。つまり、$V_n(P_i)$ は平面上で P_i が他のどの P_j よりも近いという点から成る領域である。

平面全体は $V_n(P_i)$ $(i = 1, 2, \ldots n)$ によって分割され、それをボロノイ図という。図 1 の点集合に対するボロノイ図を図 4 に示す。点 P_i $(i = 1, 2, \ldots n)$ をボロノイ図の母点という。ボロノイ多角形の頂点をボロノイ点、辺をボロノイ辺という。ボロノイ点は、通常次数が 3 であり (その点から出ている辺の数が 3)、その点に集まる 3 つのボロノイ多角形の母点の外心である。ボロノイ辺は、その両側のボロノイ多角形の母点の垂直二等分線となっている。ボロノイ図も、計算幾何学の最も基本的な問題で、平面上の施設配置をはじめとする都市計画、社会工学上の応用も数多くある。小学校の校区の妥当性の検証、商圏分析などがその応用例である。

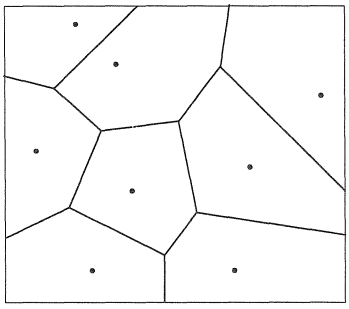

図 4　図 1 の点集合に対するボロノイ図

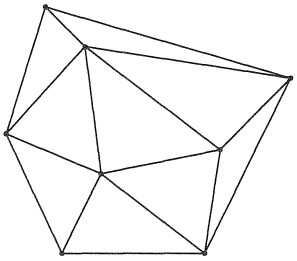

図 5: 図 1 の点集合に対するデローネ三角形分割

2.3 デローネ三角形分割

ボロノイ図においてボロノイ辺で隣接する二つのボロノイ多角形の母点を線分で結ぶと、n 個の母点の凸包が母点を頂点とする多角形に分割される。ボロノイ図を平面グラフと見ると、この双対グラフとなる。この双対グラフの各面は三角形であることからこのグラフをデローネ (Delaunay) 三角形分割という。図 1 の点集合のデローネ三角形分割は図 5 に示すとおりである。

デローネ三角形分割にはいろいろな興味深い性質がある。

1. 各三角形の外接円の内部に他の頂点が含まれない (図 6 参照)。
2. 三角形分割は同じ点集合の中では最小角最大となる三角形分割である。
3. 最小木は三角形分割に用いられている辺集合の部分集合である。

デローネ三角形分割も数多くの応用分野がある。後に紹介する三角形メッシュ生成は、その代表例である。デローネ三角形分割の構成方法で最も分かりやすいものを一つ紹介する。まず、点集合の凸包を計算した後に、凸包多角形を適当に三角形分割しておく。1 辺を共有する二つの三角形からなる四角形 ABCD に対して (1) 四角形 ABCD が非凸であるか、または (2) 四角形 ABCD が凸で、各々の三角形の外心が残りの頂点を内部に含まないとき、その対角線 (二つの三角形の共通辺) をデローネ辺という。デローネ辺でない辺を非デローネ辺という。隣接するすべての二つの三角形の共通辺がデローネ辺なら、デローネ三角形分割が得られている。そうでないなら、現在の対角線をもう一つの対角線と交換する。この操作を対角変形という。この操作を非デローネ辺がなくなるまで続けることによって、デ

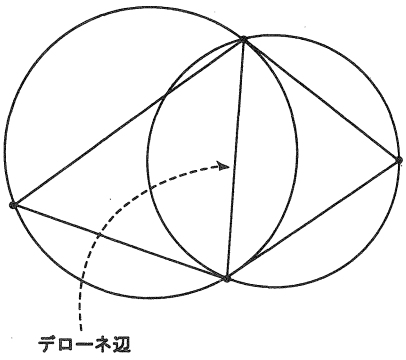

図 6　デローネ辺

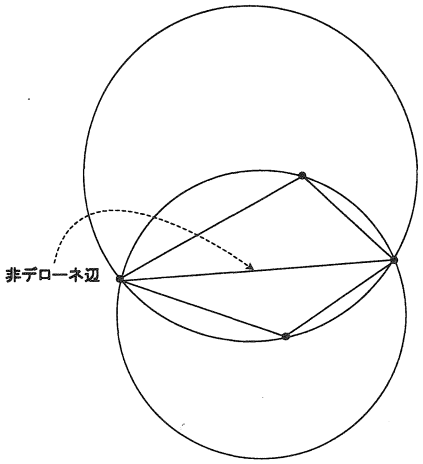

図 7　非デローネ辺

ローネ三角形分割が得られる。対角変形の操作は $O(n^2)$ 回で済むことが知られている。これは最悪の場合であって、実際にはもっと少ない回数でデローネ三角形分割がを求めることができる。

デローネ三角形分割が得られると, これを用いてボロノイ図を作ることができる。ボロノイ図はデローネ三角形分割の双対グラフであるので、デローネ三角形分割において辺によって結ばれている 2 点のボロノイ多角形は隣接している。したがってその 2 点の垂直二等分線がボロノイ辺となる。ボロノイ点は各三角形の外心である。

2.4 可視性問題

陰線処理とはコンピュータグラフィックスでよく現れる実用的な問題で立体図形をある角度から眺めたときに人間の目から見えるものだけをコンピュータの画面に映し出すための技法である。レンダリングやゲームソフトなどの基盤技術として用いられている。ここでは簡単のため，３次元のデータではなく２次元のデータを扱い，データは互いに交差しない線分の集合からなるものとする。いま，その線分集合を $\{s_1, s_2, \ldots, s_n\}$ とし，x 軸に垂直な線分はないものとする。今，視点を y 座標が $-\infty$ のある地点におき，そこから y 座標が $+\infty$ の方向を眺めるものとする。このときに，目に見える線分とその座標をリストアップする問題を考える (図 8 参照)。

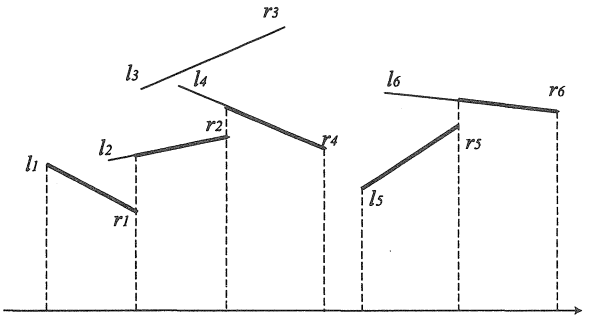

図 8　可視性問題

s_i の左（右）端点をそれぞれ $l_i(r_i)$ とし，$l_i = (x_i^l, y_i^l)$ $(r_i = (x_i^r, y_i^r))$ とする。$2n$ 点の集合 $\{l_i, r_i \mid 1 \leq i \leq n\}$ を x 座標の小さい順にソートする。それを $t(j), j = 1, 2, \ldots, 2n$ とする。ここで $t(j)$ はある $i(1 \leq i \leq n)$ に対して，$t(j) = r_i$ または $t(j) = l_i$ である。そして、$t(1)$ から順に調べていき（すなわち視線を x 軸の左から右に動かしていく），現在の視線上にある線分のなかで y 座標が最小の線分がその時点で見えている線分である。視線を動かした時に見えている線分が変化するのは、(1) 現在見えている線分の右端に達した時か、

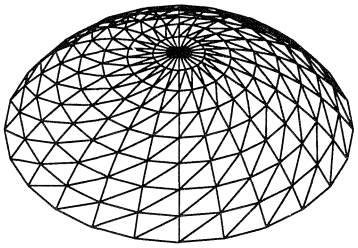

図 9　シュヴェドラードーム

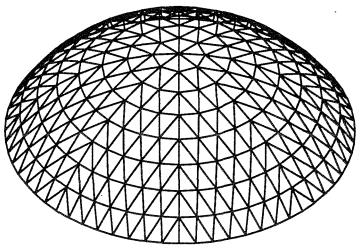

図 10　パラレルラメラドーム

(2) 新しい線分の左端に達した時である。(1) の場合は、その線分を削除して現在の視線上にある残りの線分集合のなかで y 座標が最小の線分を求めたらよい。(2) の場合は新しい線分が今まで見えていた線分より下にあるかどうかを調べて、そうであるなら見えている線分を更新する。(1) が生じると最小値計算 (現在の線分集合のサイズに比例する時間) がかかり、(2) の場合は定数時間で済む。最悪の場合は $O(n^2)$ 時間を要する。

この計算時間を改善するには優先順位付きキューというデータ構造を用いる。上述の方法の計算時間のボトルネックは (1) の場合の最小値の再計算の部分である。ある時点で視点から見て線分を近い順に並べて配列で記憶しておくとすると、最小値を削除して残りの集合に対して最小値を求めるのは、配列の 2 番目の要素を見ればよいのだが、(2) で新しい要素を追加する作業は、配列のどの位置に入るのかを計算する必要があり、配列において、追加された要素以降のすべての要素を一つずつずらす必要がある。この部分が最も時間を要する。その代わりに優先順位付きキューは要素の追加、削除、最小値の計算をいずれも $O(\log n)$ 時間でおこなう (順位付きキューの詳細は [9] を参照)。この問題では優先順位として線分の左端点の y 座標を考えてそのの小さい順に高い優先順位を与える。したがって、計算時間は全体で $O(n \log n)$ 時間である。$On \log n)$ と $O(n^2)$ の差は $n = 1000$ の場合でも 100 倍も異なるので、この違いは大きい (対数の底は省略したが 2 である)。

3. 曲面上の一様三角形メッシュ生成と立体トラス部材配置最適化への応用

空間骨組構造の設計問題では，力学的性能に関する制約条件，意匠上の制約条件のもとで最適設計を行なっている．本節ではこの設計問題から生じる球面上の一様三角形メッシュ生成に関する新しいアルゴリズムを紹介し，と立体トラス部材配置最適化への応用について言及する。

大スパンの空間を覆う屋根として用いられるトラス構造物の中で，単層の形式のトラスは，三角形のユニットで構成されることが多い (図 9, 10 参照)。

三角形ユニットからなる部材配置を決定するときの条件としては，外力作用時の応答量などの力学的特性に加えて，ユニットの形状，部材長の分散などの意匠的要求が存在し，それらを制約条件あるいは目的関数として導入した最適化問題を定式化することができる。しかし、ここでは力学的特性に関する要求や制約は陽に考慮せずに部材長のばらつきのみを考慮する問題を考える。実際, 生産コストの観点から部材長のばらつきを小さくすることが必要であり，ばらつきが小さければ得られる三角形要素は正三角形に近くなり力学的特性の観点からも優れた構造物となることが期待される。また、部材長のばらつきが十分小さいと, 意匠的にも優れた設計となることが期待される。したがって，問題は, 与えられた曲面に所与の個数の節点を配置し，曲面を辺長のばらつきの少ない三角形メッシュで近似する問題として定式化できる。

これらを背景に，筆者等 [2, 10] は, 最近次のような最適化基準の下での三角形メッシュ分割問題を考察した。

問題 1： 与えられた曲面上に一定個数の節点を配置して, その点集合の三角形分割で生じる最大辺と最小辺の長さの比を最小化する三角形メッシュ分割を求めよ。

問題 2: 配置する節点の数は固定されてなく、代わりに辺長の下限値 $\underline{l}$ が与えられたとき、所与の曲面上に適当な数の節点を配置して, その点集合の三角形分割で生じる最大辺と最小辺の長さの比を最小化する三角形メッシュ分割を求めよ。

これらの問題はNP困難と考えられる難しい問題であるが, [2] において, 平面上の凸多角形領域に対して逐次ボロノイ分割アルゴリズムにより定数近似の解が容易に得られることが示された。ここで、問題がNP困難とは、その問題を厳密に解く効率的な解法が存在しそうにないことを意味する。厳密な定義は [9] などを参照されたい。たとえば、また, [10] においてこのアルゴリズムはさらに曲面上の三角形メッシュ生成に拡張されている. 本稿では基本的な理論を平面の場合で説明し、球面上の三角形メッシュ生成について得られた実験結果を紹介する。

まずはじめに注意すべき点は、二次元平面の場合に限っても問題 1 , 2 を厳密に解くのは極めて難しいことである。これらの問題と少し異なるが、関連の問題として円の充填問題がある。同一半径の一定個数の円を定められた領域 (正方形) に重なること無しに詰め込むことができるような、円の半径の最大値を求める問題である。

他方、問題 1, 2 の一次元版は容易に解ける。たとえば線分と線分上の点集合 (線分の両端点はこの集合に入っているとする) が与えられたとき、上記の問題 1 を解くのは簡単である (詳細略)。近似解法として次のような簡単な方法 (貪欲算法) を考えてみよう。線分はその上にあらかじめ配置された点集合によって、複数の線分に分かれている。そのなかで、最大長の線分を選んでそれを二等分するように、真中に新しく点を配置する。この操作を必要な数の点が配置されるまでおこなう。たとえば、最初に得られている線分の最小長を $l_{\min}$ とする。以上の操作を続けていって、最小の線分の長さが $l_{\min}$ 未満になれば、線分の最大長と最小長の比は明らかに 2 以下になる。

[2] の結果はこの考えを二次元に拡張して、最大辺と最小辺の長さの比が 6 以下になるような三角形メッシュ分割が得られることを問題 1 に対して示した。平面上の凸多角形 $\boldsymbol{P}$ が与えられ、さらに境界上にあらかじめいくつかの節点が配置されているものとする。その方法は、はじめに、ボロノイ図を応用したボロノイ逐次分割とよばれる方法で 2 点間の距離の最小値ができるだけ大きくなるように一定個数の点を配置した後、あらかじめ配置された点集合と新たに配置した点集合の和集合に対してデローネ三角形分割を求めてそれを解とするものである。逐次ボロノイ分割は一次元の場合の最大線分を逐次 2 分割するという考え方を二次元に拡張したものであり、考え方は極めて理解しやすい。以下の節でもう少し詳しく説明をおこなう。

3.1 問題の定式化と解法

凸多角形 $\boldsymbol{P}$(境界を含む) と境界上の頂点集合 V が与えられている。平面上の二点 u, v に対して $l(u,v)$ を u, v 間のユークリッド距離とする。また、点集合 X, Y の間の最小距離を $l(X,Y) = \min\{l(x,y) \mid x \in X, y \in Y, x \neq y\}$ によって定義する。S を $\boldsymbol{P}$ の内部または境界上に配置された n 点集合とし、 V を $\boldsymbol{P}$ の頂点集合とする。すると, 上の二つの問題は次のように定式化される。

$$\textbf{問題 1:} \quad \min_{S \subset \boldsymbol{P}, |S|=n} \min_{T \in \mathcal{T}} \frac{\max_{e \in T} l(e)}{\min_{e \in T} l(e)}.$$

$$\textbf{問題 2:} \quad \min_{S \subset \boldsymbol{P}} \min_{T \in \mathcal{T}} \frac{\max_{e \in T} l(e)}{\min_{e \in T} l(e)} \quad \text{subject to} \quad l(e) \geq \underline{l}$$

ここで $\mathcal{T}$ は $S \cup V$ に対するすべての三角形分割の集合を表す。

逐次ボロノイ分割アルゴリズムは, すでに配置された点集合から最も遠い点位置を領域内から選び、そこにあたらしく点を配置するという操作を繰り返すもので、以下のアルゴリズム INCREMENT として記述される。アルゴリズムは理解しやすいが、それが一様な三角形メッシュを生成しているということを理論的に示すのはそれほど容易ではない。ここ

で、アルゴリズムの性能 (解の精度や計算速度) の理論的解析がなぜ重要なのかについて一言述べておく。言うまでもないことだが、具体的な問題に対して、そのアルゴリズムが、最悪の場合でもどのくらいの計算時間で、厳密な解 (最適解) からどのくらい離れている解を出力するのかということを理論的に保証することは利用者にとって重要であることは明白であろう。世の中のアルゴリズムにはこのような保証のないものも多数存在する。遺伝的アルゴリズムはその代表的な例である。これはこれで実用的価値は高いが、解の精度については何も保証していない。もう一つ重要な点は、理論的に優れたアルゴリズムは実際的にも優れたアルゴリズムであるか、実際的に優れたアルゴリズムの開発の鍵となるアイデアを提供していることである。

以下では問題 2 に対する逐次ボロノイ分割アルゴリズムについて説明する。まず最初に $l(V,V) \geq 2\underline{l}$ が満たされていると仮定する。つまり、最初に配置されている点集合の間の距離は十分大きいと仮定する。問題 1 の解法については [2, 10] を参照されたい。

Algorithm INCREMENT

Step 1 $\boldsymbol{P}$ の境界上に点を配置して (配置した点集合を B とする)、$\underline{l} \leq l(V \cup B, V \cup B) \leq \beta\underline{l}$ となるようにする (ただし、$\beta \leq 3$)。$V' = V \cup B$ とする。

Step 2 $S := \emptyset$ とする。

Step 3 $\boldsymbol{P}$ 内で $l(x, V' \cup S)$ を最大にする点 p^* を求める。そのために $V \cup S$ に対する二次元ボロノイ図 $\mathrm{Vor}(V' \cup S)$ を利用する. $\mathrm{Vor}(V' \cup S)$ のボロノイ頂点の各点 p に対して、もし p が $\boldsymbol{P}$ の外部にあるなら, p を端点とするボロノイ辺と $\boldsymbol{P}$ の辺との交点 p' を求める。$\boldsymbol{P}$ の内部のボロノイ頂点と上で求めた $\boldsymbol{P}$ の辺との交点 p' の集合のなかで p^* を $V \cup S$ からの最遠点とする。

Step 4 $l(p^*, S \cup V') \geq \underline{l}$ なら、$S := S \cup \{p^*\}$ とし、Step 3 に戻る。$l(p^*, S \cup V') < \underline{l}$ なら、Step 5 へ行く。

Step 5 $S \cup V'$ の点集合に対して、デローネ三角形分割を解として出力する。

Step 1 であるが、$l(V,V) \geq 2\underline{l}$ の仮定より、常に $\underline{l} \leq l(V \cup B, V \cup B) \leq 3\underline{l}$ となるように配置できる (詳細略)。とくに、$\boldsymbol{P}$ のどの内角も 60° 以上なら、常に $\underline{l} \leq l(V \cup B, V \cup B) \leq 2\underline{l}$ となるように配置できる。その理由は以下の通りである。まず、$\boldsymbol{P}$ のどの内角も 60° 以上であるので、$l(V \cup B, V \cup B)$ を実現する 2 点は境界上で隣接する 2 点であることに注意しておく。また、$l(V,V) \geq 2\underline{l}$ の仮定より、$\boldsymbol{P}$ の辺上に点を配置して辺上で隣接する 2 点間の距離が $\underline{l}$ 以上、$2\underline{l}$ 以下にできる。

図 11 は点 p^* が選ばれていく様子を表している。ただし、$B = \emptyset$ である。

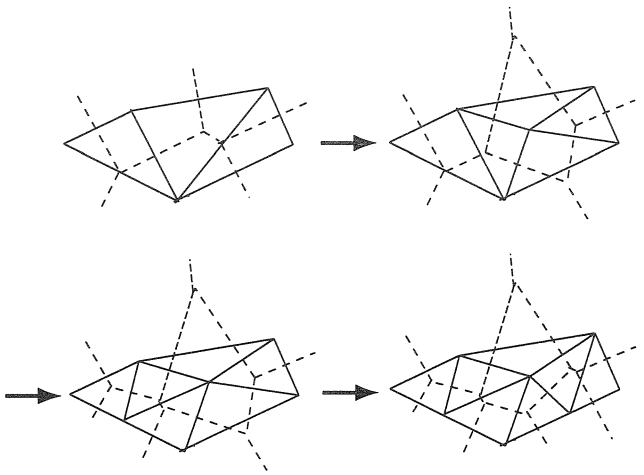

図 11　逐次ボロノイ分割

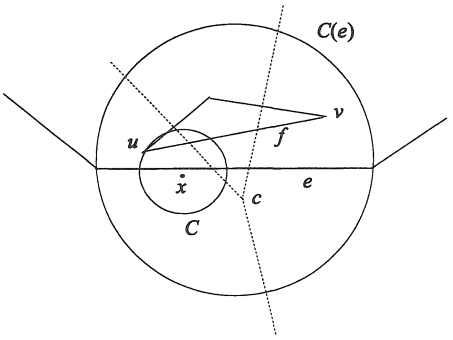

図 12　Δ の外心が $\boldsymbol{P}$ の外部にある場合

定理 1 $\alpha = \max\{2, \beta\}$ とする。アルゴリズム *INSERT* によって得られた S_n は上記の問題 *(2)* の α-近似解である。

証明. 最終的に選ばれた S の要素数を n とする。$S \cup V'$ のデローネ三角形分割 $DT(S \cup V')$ を考える。$DT(S \cup V')$ の一つの三角形 Δ を考える。Δ の外心を c とする。$\boldsymbol{P}$ が c を含むかどうかによって場合分けして考える。

Case 1: $\boldsymbol{P}$ が c を含む場合。Δ の一辺 $e = (u, v)$ を考える。アルゴリズムの Step 4 で最後に選ばれた点を p_n^* とする。p_n^* が選ばれた後、Step 3 に戻り、$l(x, S \cup V')$ を最大にする点 p^* を選ぶ。この点を p_{n+1}^* とする。しかし、$l(p_{n+1}^*, S \cup V') < \underline{l}$ であったので S に追加されずに Step 5 に行き、終了した。p_{n+1}^* は $l(x, S \cup V)$ を最大にしているので、$l(c, u) \leq l(p_{n+1}^*, S \cup V')$ である。よって、$l(c, u) < \underline{l}$ である。同様に、$l(c, v) < \underline{l}$ である。よって、距離の三角不等式より、$l(u, v) < 2\underline{l}$ となる。

Case 2: Δ の一辺 $f = (u, v)$ を考える。このとき、$S \cup V'$ に対するボロノイ図 $\mathrm{Vor}(S \cup V')$ の母点 u のボロノイ領域は $\boldsymbol{P}$ の一つの境界辺 e と交わる (図参照)。

u のボロノイ領域にあって $\boldsymbol{P}$ の外部にある点 x を選び、x を中心とし、u を含み e の両端点を含まない円 C を考える。すると、そのような円 C は e を直径とする円 $C(e)$ に完全に含まれる。したがって、u は $C(e)$ に含まれる。同様に v も $C(e)$ に含まれる。以上から $l(f) \leq l(e)$ を得る。よって、$l(f) \leq \beta\underline{l}$ が成り立つ。

以上のことと、$DT(S \cup V')$ のどの辺の長さも $\underline{l}$ 以上であることから、定理が成り立つ。□

問題 1 の場合は、アルゴリズムが少し異なる。まず、境界上に点を配置せずに逐次ボロノイ分割アルゴリズムを走らせて、$l(S, S \cup V)$ を求めておく。この情報をもとに、境界上に適当に点を配置した後、再び逐次ボロノイ分割アルゴリズムを走らせて、三角形分割を得るという方法である。ただし、最大辺と最小辺の長さの比の上限は 6 となり、問題 1 に比べて悪くなり、解析も複雑である。

曲面上の三角形メッシュ生成も二次元の場合とほぼ同様におこなわれる。$l(x, S \cup V)$ を最大にする点を曲面上で求める必要がある。最終的に出力する三角形メッシュは、たとえば半球面の場合、上側凸包である。最大辺と最小辺の長さの比は二次元の場合と変わらない。

3.2 実験結果

問題 1 に対して、半球面や球を平面で切り落とした領域 (半球面より小さいものとする) に対して色々実験を行ったが，最大辺の長さと最小辺の長さの比は常に 2 以下でり、上記の理論的評価よりかなり優れていた。計算実験からすると、提案手法は実用的観点からも優れた方法と言える。

以下, 実験結果を示す。13 に示すような境界に等間隔に 4 つの初期点が与えられた半球面を考える。$n = 50, 100, 200, 300, 400, 500$ の場合に逐次ボロノイアルゴリズムを適用すると，いずれの場合も辺長比は $1.9 \sim 2.0$ の間に収まった。$n = 100, 500$ の場合に得られた解

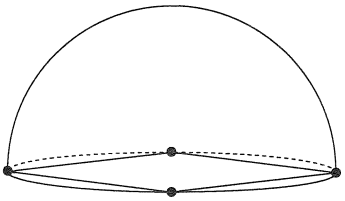

図 13　半球面と初期点

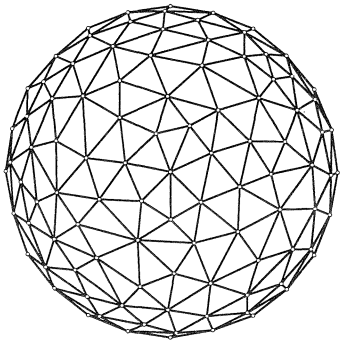

図 14　得られた三角形メッシュ($n = 100$)

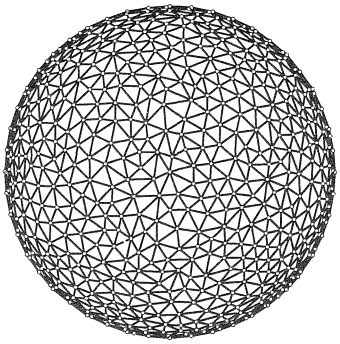

図 15　得られた三角形メッシュ($n = 500$)

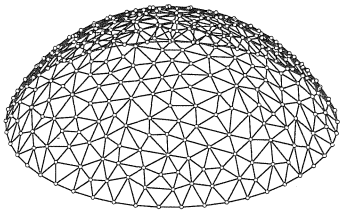

図 16: 半球面より小さい部分球面における三角形メッシュ($n = 300$)

を図 14 図 15 に示す。半球面よりも小さい部分球面に対する実験もおこない，同様の成果を得ている (図 16 参照).

4. おわりに

本稿では、計算幾何学の紹介と、その技術を利用した曲面上の一様な三角形メッシュ生成アルゴリズムを紹介した。紙面の都合上、計算幾何学全般を紹介する余裕はなかったが、建築のさまざまな場面での応用価値は高い技術と思われ、読者の方も関心を持っていただけたら幸いである。

参考文献

[1] 浅野哲夫, 「計算幾何学」 朝倉書店 1990

[2] F. Aurenhammer, N. Katoh, H. Kojima, M. Ohsaki and Y. Xu, Approximating Uniform Triangular Meshes in Polygons, *Theoretical Computer Science*, Vol. 289, No. 2, (2002) 879-895.

[3] D. Avis, 今井浩, 松永信介,「 計算幾何学・離散幾何学」 朝倉書店 1994

[4] Mark De Berg, Marc Van Kreveld, Mark Overmars, Otfried Schwarzkopf, *Computational Geometry: Algorithms and Applications*, Springer Verlag, 1997, 「コンピュータ・ジオメトリ—計算幾何学:アルゴリズムと応用」, (浅野哲夫 訳) 近代科学社, 2000

[5] H. Edelsbrunner, Algorithms in Combinatorial Geometry, Springer Verlag 1987, 「組合せ幾何のアルゴリズム」(今井浩, 今井桂子 訳) 共立出版 1995

[6] R.L. Graham, An Efficient Algorithm for Determining the Convex Hull of a Finite Planar Point Set, Information Processing Letters, Vol.1, 132-133.

[7] 今井浩, 今井桂子, 「計算幾何学」 共立出版 1994

[8] 伊理正夫監修 「計算幾何学と地理情報処理 第 2 版」 共立出版 1993

[9] 岩野和生, 加藤直樹, 永持 仁, 「アルゴリズム入門 – 設計と解析 –」(翻訳), ピアソン・エデュケーション 2002 年 11 月 (原著: S. Baase, *Computer Algorithms: Introduction to Design and Analysis*, Addison-Wesley, 1989).

[10] Naoki Katoh, Hiromichi Kojima and Ryo Taniguchi, Approximating Uniform Triangular Meshes on Spheres, Proc. of JCDCG 2000, Lecture Notes in Computer Science (LNCS) Vol. 2098, pp.192-204, Springer-Verlag, Berlin, 2001.

[11] F. Preparata and I. Shamos, Computational Geometry: An Introduction, Springer-Verlag 1985, 「計算幾何学入門」(浅野孝夫 浅野哲夫訳) 総研出版 1992

[12] 徳山 豪, 「はみだし幾何学」 岩波 科学ライブラリー 1995

[13] Geometry in Action, http://www.ics.uci.edu/~eppstein/geom.html