修 士 論 文

ハードウェア支援による
各種ボロノイ図の高速算法とその応用

北陸先端科学技術大学院大学
情報科学研究科情報処理学専攻

寺本幸生

2003 年 3 月

修 士 論 文

ハードウェア支援による
各種ボロノイ図の高速算法とその応用

指導教官　浅野哲夫 教授

審査委員主査　浅野哲夫 教授
審査委員　金子峰雄 教授
審査委員　中野浩嗣 助教授

北陸先端科学技術大学院大学
情報科学研究科情報処理学専攻

110085 寺本幸生

提出年月: 2003 年 2 月

概 要

アルゴリズムの一部に特定の機能をもつハードウェアを導入し，高速化を図る技法をハードウェア支援技法という．近年，グラフィックスアクセラレータを用いて，離散的なボロノイ図を高速に求める手法が提案された．本論文では，このハードウェア支援技法で得られるボロノイ図を用いて等高線地図の補間を高速に計算するアルゴリズムを提案する．補助等高線は，乗法重み付き距離での一般化ボロノイ境界を検出することで得られる．また，いくつかの一般化ボロノイ図を定式化し，その応用性について議論する．

目 次

第 1 章　はじめに **1**
1.1　研究の背景 1
1.2　研究の目的 4

第 2 章　ボロノイ図 **6**
2.1　普通のボロノイ図 6
2.2　離散ボロノイ図 7
2.3　一般化ボロノイ図の定義 9
2.3.1　d 次元空間における一般化 9
2.3.2　サイトの形状における一般化 9
2.3.3　距離関数における一般化 10
2.3.4　その他の観点における一般化 11

第 3 章　ハードウェア支援技法 **14**
3.1　基本概念 14
3.2　隠面除去によるボロノイ領域の表現 15
3.3　各サイトの距離関数メッシュ 15
3.4　ハードウェア支援技法による誤差 16
3.4.1　距離関数近似による誤差 16
3.4.2　空間のサンプリングによる誤差 17

第 4 章　計算コストの比較 **18**

第 5 章　いろいろなボロノイ図 **20**
5.1　L_1, L_∞ 距離でのボロノイ図 20
5.2　楕円距離のボロノイ図 22
5.3　加法，乗法，複合重みのボロノイ図 24
5.4　Hausdorff 距離のボロノイ図 25
5.5　その他 26

第 6 章　等高線地図の補間 **27**
6.1　乗法重み付きボロノイ図 27

6.2 ハードウェア支援技法による実現 28
6.3 実行結果 29
6.4 提案アルゴリズムの漸近的解析 29

第 7 章 今後の課題 **31**

第1章　はじめに

1.1　研究の背景

計算幾何学（*computational geometry*）は，幾何学的な問題を計算機を用いて解決する方法を研究する学問であり，近年の計算機の能力向上にともなって急速に発展している研究分野である．ボロノイ図（*Voronoi diagram*）に関する研究は，計算幾何学における主要なテーマの一つであり，情報科学においては地理情報処理やコンピュータグラフィクスなど多くの分野への応用がなされている．

計算幾何学は1970年代初期に起こった研究分野であるが，ボロノイ図の歴史はそれよりも古く，既に少なくとも4世紀間にわたって研究がなされている．ボロノイ図がこれほどまでに長く継続して研究されてきた背景には，ボロノイ図およびその双対として知られるドローネ図が持つ特性が，情報科学に留まらず非常に多くの分野に寄与していることが関係している．

ボロノイ図を構成するためのアプローチとして，様々なパラダイムが計算幾何学の分野で考案された．2次元のユークリッド空間 $\mathbf{E}^2$ における n 点のボロノイ図は，逐次点加法（*incremental method*），分割統治法（*divide-and-conqure*）によりそれぞれ最悪の場合 $O(n^2)$，$O(n \log n)$ で計算することができ，入力点集合が一様に分布されているような平均の場合では共に $O(n)$ で構成することができる．特に，逐次添加法は一様分布していないような一般的な入力であっても，実用的には $O(n)$ に近い計算時間で実行できることが経験的に分かっている．他にも最適なアルゴリズムとして，平面走査法（*plane sweep method*）[9] などがある．また，多次元のサイト集合に対して効率よくボロノイ図を構成するためのパラダイムとして，リフトアップ法（*lift-up method*）[10, 11] がある．この技法は，$\mathbf{E}^d$ 上の点 $\mathbf{p} = (x_1, x_2, \ldots, x_d)$ を $\mathbf{E}^{d+1}$ 上の点 $\mathbf{p}^* = (x_1, \ldots, x_d, x_1^2 + \cdots + x_d^2)$ に変換し，$d+1$ 次元凸包を計算する．この $d+1$ 次元凸包を元の $\mathbf{E}^d$ 次元平面に射影して得られる図形から d 次元ボロノイ図を構成する．$d+1$ 次元凸包は，分割統治法により $O(n \log n + n^{\lfloor \frac{d}{2} \rfloor})$ 時間で計算でき，d 次元ボロノイ図の組合せ的複雑度は $O(n^{\lfloor \frac{d}{2} \rfloor})$ であるからこの手法は最適に d 次元ボロノイ図を構成することができる．

実用的なボロノイ図の構成アルゴリズムに関しては，もう少し厳密な議論を要する．理論的に正しいアルゴリズムであるという証明があったとしても，実用的に妥当なプログラムを実装できるという保証はない．即ち，縮退（*degeneracy*）や数値誤差（*numerical error*）に対する頑健性（*robustness*）を考慮しなければならない．幾何学的アルゴリズムにおける判定は，位相判定と数値判定に分けることができる．位相判定は離散的な情報で

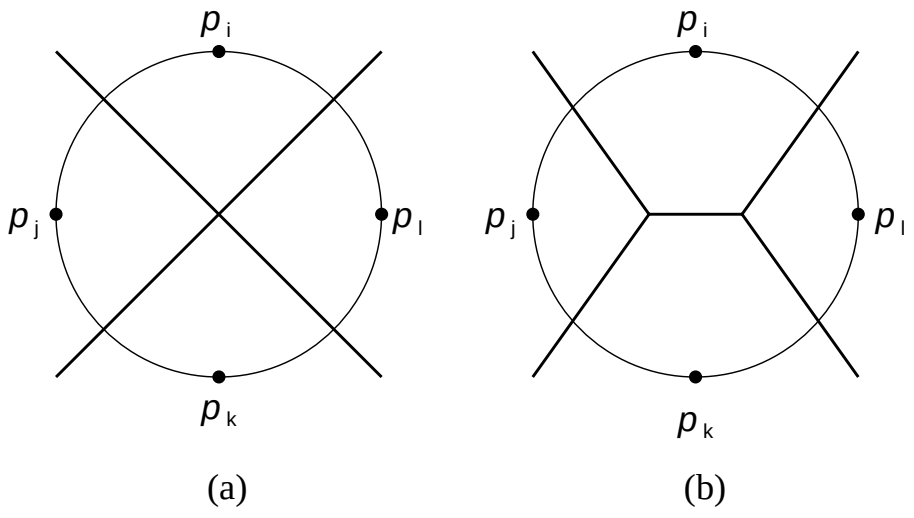

図 1.1: 縮退による位相矛盾: (a) 正しいボロノイ図，(b) 縮退により隣接関係が変わってしまう例.

ある位相構造に対する判定であるため，計算機によって正確に真か偽かの判定を行うことができる．これに対して数値判定は，一般に連続量に対する判定式の符号が正であるか負であるかの判定であるが，判定式の値は正負の他に 0 である場合がある．入力データが縮退であるというのは，数値判定において 0 を判定してしまうような特別な場合をいう．例えば，ボロノイ図の計算では，4 点以上の点が同一円周上にありその内部に他の入力点を含まない場合に縮退となる（図 1.1）．即ち，式 1.1 において $F(\mathbf{p}_i, \mathbf{p}_j, \mathbf{p}_k, \mathbf{p}_l) = 0$ となる場合である．

$$F(\mathbf{p}_i, \mathbf{p}_j, \mathbf{p}_k, \mathbf{p}_l) \equiv \begin{vmatrix} 1 & x_i & y_i & x_i^2 + y_i^2 \\ 1 & x_j & y_j & x_j^2 + y_j^2 \\ 1 & x_k & y_k & x_k^2 + y_k^2 \\ 1 & x_l & y_l & x_l^2 + y_l^2 \end{vmatrix} \tag{1.1}$$

ただし，点 $\mathbf{p}_i = (x_i, y_i)$ とする．

図 1.1 のように，縮退となる入力データが与えられたとき，位相矛盾（*topological inconsistency*）は数値誤差によって引き起こされる．つまり，三角形 $\mathbf{p}_i, \mathbf{p}_j, \mathbf{p}_k$ の外接円の中心と，三角形 $\mathbf{p}_i, \mathbf{p}_k, \mathbf{p}_l$ の外接円の中心が数値誤差により一致しない場合である（図 1.1(b)）．これは，浮動少数点を用いた演算（*floating-point arithmetic*）による，丸め誤差（*rounding error*）が引き起こす判定の歪みに起因する．また，固定ビット長での演算は深刻なプログラムの破綻を導く恐れがある．図 1.2 は，逐次添加法における手続きで数値誤差によりプログラムが暴走する例を示している．黒点および実線は，既に加えられた点とそのボロノイ図を表しており，白点および破線は新たに加えられた点とそれによって構成されるボロノイ境界をそれぞれ表している．このとき点 p, q は一致していなければならないが，数値誤差が積み重なり閉じていない境界を得る．さらに，ボロノイ境界の形成手続きは停止せ

ず，次のボロノイ辺を計算しようとする．これにより，プログラムは無限ループに陥る．

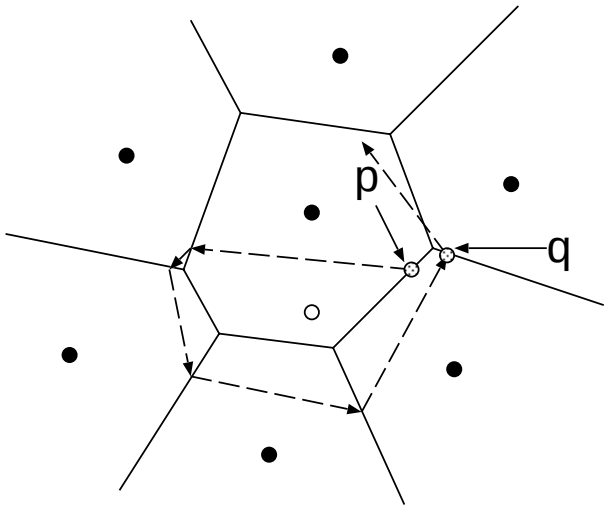

図 1.2: 数値誤差により閉じていない境界を出力してしまう例

このような縮退や数値誤差に起因する問題に対して，頑健なアルゴリズムを構築する技法がいくつかある．幾何学的レベルでは，位相優先法（*topology-oriented approach*），記号摂動法（*symboric perturbation*）などがあり，演算レベルでは，誤差無し演算（*exact arithmetic*），整数帰着法などがある．

位相優先法は，位相構造における無矛盾性を優先し，数値判定の使用を最小限にとどめることによって，数値判定がどのように誤っても位相的な矛盾を生じない算法に直す技法である [7, 8]．この技法が適用されたパラダイムとして，逐次添加法 [12] や分割統治法 [13] が知られている．

記号摂動法は，入力データの数値をわずかに変更することによって縮退を回避する手法である．これにより，設計と実装の段階で特殊な場合を無視しても正しく動作するようにできる．この技法を用いる場合，点の座標は整数でなければならないため整数帰着法を併用しなければならない．整数帰着法は，有限ビット長で与えられた入力値を十分大きな定数をかけることによって全て整数値に近似し，整数演算によって誤差無しで計算を行う技法である．記号摂動法および整数帰着法は，誤差対策に関して単純かつ強力なアプローチといえる．

また，計算機の能力向上にともない，誤差無し演算に対するアプローチも現実的なものになってきた．多倍長整数演算のためのライブラリとして，LEDA の real データタイプ [15, 14] や CORE[16] ライブラリなども実用的なものとして広く普及し始めている．

このように，幾何学的アルゴリズムが持つ特有の問題を考慮することは，実用的な面で非常に大切なことといえる．一方で，ある一定の精度保証が得られるよう空間をサンプリングすることにより，連続空間を離散的に表現し，その離散空間内でボロノイ図を定義する研究も実用的な側面から進められている．例えば，コンピュータグラフィクスでのディ

ジタル画像の扱いに関して，その画像の解像度に依存するようなアルゴリズムを考える場合，連続空間としてとらえるよりも離散空間としてとらえる方が妥当である．離散的なボロノイ図（*discrete Voronoi diagram*）は，各サンプリング点がどのサイトに最も近いかによって空間を分割した図形であり，コンピュータグラフィクス，CAD/CAM などの分野で応用されている．

近年，離散的なボロノイ図をハードウェア的な戦略で扱う新たなパラダイムが提案された [1]．この技法はハードウェア支援技法（*hardware-assisted techniques*）と呼ばれ，最近特に注目されている．ハードウェア支援技法は，アルゴリズムの一部に特定の機能を持つハードウェアを導入し，高速化を図る技法である．離散的なボロノイ図の計算に対してはグラフィクスハードウェア（*graphics hardware*）の導入により計算の高速化が可能である．この技法は，高速であるだけでなく幾何問題がもつ縮退や計算誤差に対して非常に頑健であり，実装が容易であるという特徴をもつ．また，得られるボロノイ図はディジタル画像の形式であるため，計算幾何学的手法で得られる情報に加えてディジタル画像特有の情報をも取得することができる．これらの利点を活かして，様々な応用分野で用いられ始めている [2, 3]．

1.2 研究の目的

本研究は，ハードウェア支援技法によるボロノイ図の高速算法に着目し，この技法の高速性を利用した応用について調査することを目的とする．この技法の主要な特徴として，ハードウェアの介在によるシステムの高速化，および OpenGL などグラフィックスライブラリによる実装の簡便さ，データの透過性があげられる．

グラフィックスハードウェアによるボロノイ図算法は，ある定義に基づく 3 次元の距離関数をシーンに描画することによってなされる．この距離関数を操作することによって，様々な一般化ボロノイ図（*generalized Voronoi diagram*）を構成することができる．本論文では一般化ボロノイ図として，ミンコフスキー距離（*Minkowski metric*）に基づくボロノイ図，楕円距離のボロノイ図（*elliptic distance Voronoi diagram*），重み付き距離のボロノイ図（*weighted distance Voronoi diagram*），そしてハウスドルフ距離のボロノイ図（*Hausdorff distance Voronoi diagram*）を定式化する．また，これらの一般化ボロノイ図の応用例について考察する．

この技法で求められるボロノイ図は，計算幾何学的手法で得られるボロノイ図の情報とは異なり，ディジタル画像の形式で得られる．そのため，ボロノイ頂点やボロノイ辺の集合を抽出したり，ボロノイ領域の隣接関係などの情報は後処理をしなければ知ることができない．この後処理は，大きさが $n \times m$ のシーンに対し $O(nm)$ の計算時間を必要とするため，ハードウェア支援技法を用いる利点を損なう．

しかし，ディジタル画像形式のボロノイ図表現がそのまま利用できる応用面も多数ある．グラフィックスハードウェアの機能にも依存するが，例えば，空間のメトリック変換や，あるボロノイ領域の面積計算などの処理は，$O(1)$ 時間で得ることができる．さらに，

ハードウェア支援技法によるディジタル画像の加工を行うような問題にも適しているといえる．このように，ディジタル画像形式のボロノイ図表現を用いて効率よく解ける問題と，ボロノイ図のもつ幾何学的情報を用いないと解けない問題に区別することができる．

ハードウェア支援技法が柔軟に介在できる応用例として，等高線地図の補間に関する問題を対象とする．等高線地図の補間は，データの無い地域の標高を推定し補助等高線を付加することである．一般に，精密的方法と近似的方法に区別されるが，情報科学的な観点から後者のデータを平滑する方法に基づきハードウェア支援技法を適用する．入力として与えられる等高線集合，つまり自己交差の無い多角形チェイン（*polygonal chain*）の集合に関して，各等高線をサイト（*site*）とするボロノイ図の境界はその補助等高線となる．また，重みの付けられた距離関数による一般化ボロノイ図を構成することにより，複数の補助等高線を高速に得るためのアルゴリズムを提案する．

第2章　ボロノイ図

ボロノイ図は，最近点問題を解くために用いられるデータ構造であり，与えられたサイト集合に対する勢力圏図として知られている [5]．本章では，一般化ボロノイ図の導入および，様々な一般化の観点を紹介する．まず始めに，2 次元ユークリッド平面における点集合に関するボロノイ図を定義し，その性質を示す．このボロノイ図は，一般化ボロノイ図に対して普通のボロノイ図（*ordinary Voronoi diagram*）と呼ばれる．次に，一般化ボロノイ図を定義し，いくつかの一般化の観点を示す．

2.1　普通のボロノイ図

ボロノイ図の基本的な性質を考察するために，普通のボロノイ図についてみていく．図 2.1 に，普通のボロノイ図を例示する．

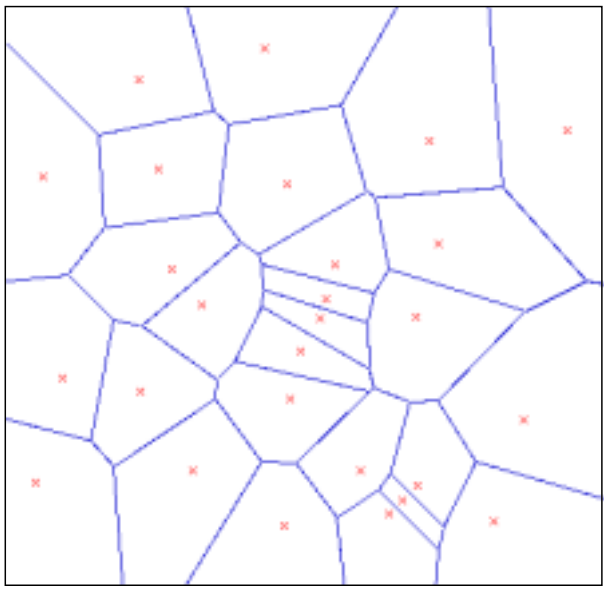

図 2.1: 普通のボロノイ図

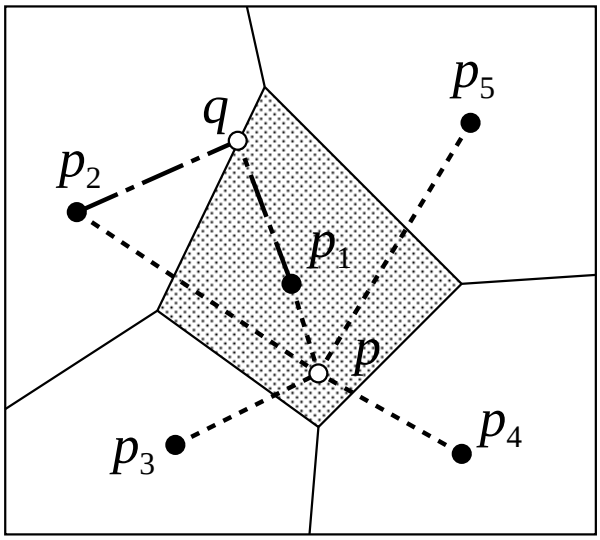

図 2.2: 平面上の点のメンバー割り当て

普通のボロノイ図の直観的な定義は，互いに素な有限の点集合が与えられたとき，平面上の全ての点を最も近い点集合の各メンバーに関連付けることで得られる領域分割である．図 2.2 は，与えられた点集合 $\{p_1, p_2, \cdots, p_5\}$ に対して平面上の点 p, q が，どのメン

バーに関連付けられるかを示している．同図において，網かけされた部分を p_1 の支配領域とすると，支配領域内の点 p は，p_1 に関連付けれる．同様に，その支配領域の境界の点 q は p_1 に関連付けられることになるが，この場合隣接する支配領域を持つメンバー q にも関連付けられる．

もう少し厳密な定義を示す．n $(2 \leq n < \infty)$ を 2 次元ユークリッド平面 $\mathbf{E}^2$ 上に与えられた点集合の大きさとする．この n 点はそれぞれ $p_1, \ldots, p_n$ によって識別され，デカルト座標 $(x_{11}, x_{12}), \ldots, (x_{n1}, x_{n2})$ もしくは，位置ベクトル $\mathbf{x}_1, \ldots, \mathbf{x}_n$ で表される．n 点が互いに素であるというのは，$i \neq j,\ i, j \in I_n = \{1, \ldots, n\}$ に関して，$\mathbf{x}_i \neq \mathbf{x}_j$ と書ける．平面上の任意の点 p を座標 (x_1, x_2) もしくは，位置ベクトル $\mathbf{x}$ で表すとき，p から p_i へのユークリッド距離を $d(p, p_i) = \|\mathbf{x}_i - \mathbf{x}\| = \sqrt{(x_{i1} - x_1)^2 + (x_{i2} - x_2)^2}$ とする．p_i が p の最も近い点か，最も近い点の一つであるとき，$i \neq j,\ i, j \in I_n$ に関して $\|\mathbf{x}_i - \mathbf{x}\| \leq \|\mathbf{x}_j - \mathbf{x}\|$ の関係を持つ．

Definition 2.1 $P = \{p_1, \ldots, p_n\} \subset \mathbf{E}^2$ を n 点の集合とする．ただし，$2 \leq n < \infty$ かつ $i \neq j,\ i, j \in I_n$ に関して $\mathbf{x}_i \neq \mathbf{x}_j$ とする．

$$V(p_i) = \{\mathbf{x} \mid \|\mathbf{x}_i - \mathbf{x}\| \leq \|\mathbf{x}_j - \mathbf{x}\|,\ j \neq i,\ j \in I_n\} \tag{2.1}$$

で与えられる領域を p_i に関連付けられたボロノイ領域（*Voronoi resion*）もしくは p_i のボロノイ領域と呼ぶ．そして，

$$\mathcal{V}(P) = \{V(p_1), \ldots, V(p_n)\} \tag{2.2}$$

によって与えられる集合を P によって生成されたボロノイ図，もしくは P のボロノイ図という．$V(p_i)$ の p_i は，i 番目のボロノイ領域のサイトと呼び，$P = \{p_1, \ldots, p_n\}$ はボロノイ図 $\mathcal{V}(P)$ のサイト集合と呼ぶ． □

ボロノイ領域は閉じているので，境界を有する．この境界をボロノイ境界（*Voronoi boundary*）と呼び，サイト p_i のボロノイ境界を $\partial V(p_i)$ と書く．ボロノイ境界は，式 2.1 より対応する 2 つのサイトの垂直二等分線，即ち線分，半直線，もしくは直線で形成される．これらのボロノイ境界を形成している部品をボロノイ辺（*Voronoi edge*）と呼ぶ．数学的に記述すると，$V(p_i) \cap V(p_j) \neq \emptyset$ なら $V(p_i) \cap V(p_j)$ はボロノイ辺を与える．ただし，この定義ではボロノイ辺は点に退化している場合も想定している．p_i と p_j によって生成されたボロノイ辺を $e(p_i, p_j)$ と表記する．任意のサイト p_i, p_j に対して，$e(p_i, p_j)$ は空となることもあり，$e(p_i, p_j)$ が空でもなく点でもない場合，ボロノイ領域 $V(p_i), V(p_j)$ は隣接（*adjacent*）しているという．ボロノイ辺の端点をボロノイ頂点（*Voronoi vertex*）と呼び，v_i と表記する．ボロノイ頂点は，少なくとも 3 つのボロノイ辺が接続している点である．

2.2 離散ボロノイ図

本節では，離散ボロノイ図の定義を行う．離散ボロノイ図は，連続平面上の点をセルと呼ばれる矩形領域で一様にサンプリングすることによって得られる離散平面上で定義され

る．離散的な点としてサンプリングする際，連続平面を与えられたサイト集合を含む十分大きな矩形領域で近似する．

Definition 2.2 $C = \{c_{ij},\ i = 1, \ldots, n_w,\ j = 1, \ldots, n_h\}$ をセルの集合，即ち離散平面上の点の集合とし，$P_c = \{p_{c1}, p_{c2}, \ldots, p_{cn}\}$ を n 個のサイト集合とする．ただし，$P_c \subset C,\ 2 \leq n \leq n_w n_h < \infty$ とする．セル c_{ij} からサイト p_{ck} への距離 $d(c_{ij}, p_{ck})$ を c_{ij} の中心の点から p_{ck} の中心の点へのユークリッド距離と定義する．このとき，サイト p_{ci} のボロノイ領域 $V_c(p_{ci})$ を

$$\begin{aligned} V_c(p_{ci}) \quad = \quad & \{c_{kl} \mid d(c_{kl}, p_{ci}) < d(c_{kl}, p_{cj}) \quad \text{for } j \neq i; \\ & d(c_{kl}, p_{ci}) = d(c_{kl}, p_{cj}) < d(c_{kl}, p_{cm}) \\ & \text{for } i < j \neq m, k \in I_{n_w}, l \in I_{n_h}\} \end{aligned} \tag{2.3}$$

のように定義する．また，離散ボロノイ図 $\mathcal{V}_c(P_c)$ を

$$\mathcal{V}_c(P_c) = \{V_c(p_{c1}), \ldots, V_c(p_{cn})\} \tag{2.4}$$

と定義する． □

連続平面におけるボロノイ図との違いは，あるセルへの距離が最小となるサイトが複数ある場合，そのセルがサイトのインデックスの小さい方のボロノイ領域に属するという点である．これは，セルの中心の点があるボロノイ辺上にある場合に起こる．図 2.3 は，離散ボロノイ図の例を示している．

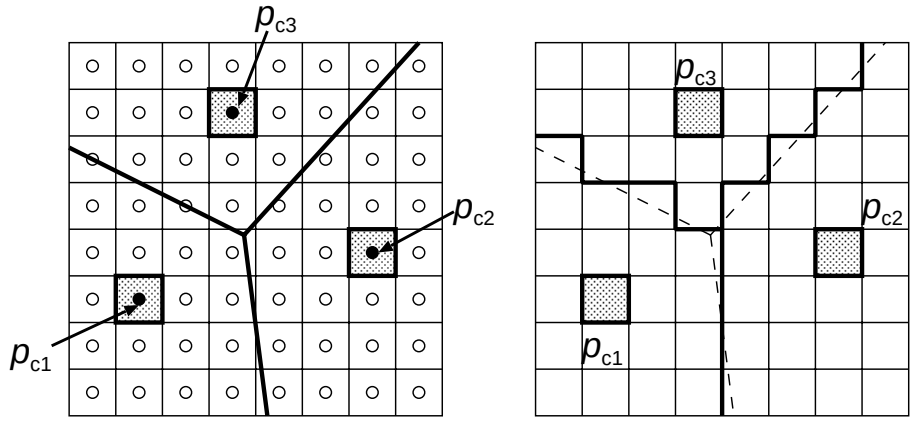

図 2.3: 離散ボロノイ図

2.3 一般化ボロノイ図の定義

$S = \{s_1, \ldots, s_n\}$ を $\mathbf{E}^d$ の互いに素な n 個のサイトの集合とする．次式によって定義される領域を s_i に関するボロノイ領域と呼ぶ．ただし，$dist$ は適当な距離関数とする．

$$V(s_i) = \bigcap_{j \neq i} \{\mathbf{p} \in \mathbf{E}^d \mid dist(\mathbf{p}, s_i) \leq dist(\mathbf{p}, s_j)\} \tag{2.5}$$

そして，次式によって与えられる集合を S によって生成されたボロノイ図という．

$$\mathcal{V}(S) = \{V(s_1), \ldots, V(s_n)\}$$

一般化ボロノイ図は，d 次元空間，サイトの形状，距離関数などをある応用問題に則して定義しなおされたボロノイ図である．

2.3.1 d 次元空間における一般化

$P = \{p_1, p_2, \ldots, p_n\}$ を $\mathbf{E}^d$ の互いに素な点集合とする．$\mathbf{E}^d$ 上のある点 $p = (x_1, \ldots, x_d)$ から，与えられた点集合の点 $p_i = (x_{i1}, \ldots, x_{id})$ への距離は，

$$d(p, p_i) = \sqrt{(x_{i1} - x_1)^2 + (x_{i2} - x_2)^2 + \cdots + (x_{id} - x_d)^2}$$

と書ける．従って，d 次元におけるボロノイ領域 $V(p_i)$，ボロノイ図 $\mathcal{V}$ は，それぞれ次のように書ける．

$$\begin{aligned} V(p_i) &= \{p \in E^d \mid d(p, p_i) \leq d(p, p_j), \quad i \neq j, i, j \in I_n\} \\ \mathcal{V}(P) &= \{V(p_1), \ldots, V(p_n)\} \end{aligned}$$

ボロノイ領域は d 次元凸多面体となるため，$d-1$ 次元以下の超平面によって境界付けられる．例えば 3 次元ボロノイ図の場合，ボロノイ境界は平面，線分，そして点によって構成される．図 2.4 は，３次元ボロノイ図の例を示している．

2.3.2 サイトの形状における一般化

$S = \{s_1, s_2, \ldots, s_n\}$ を $\mathbf{E}^2$ の互いに素なサイト集合とする．サイト集合の任意のサイトは点とは限らないことを仮定する．サイトの形状として，点，線分，円弧，多角形などが考えられる．このとき $\mathbf{E}^2$ 上の任意の点からサイト s_i への距離は，

$$d(p, s_i) = \min_{\mathbf{x}_i} \{\|\mathbf{x} - \mathbf{x_i}\| \mid \mathbf{x}_i \in s_i\}$$

のように表せる．図 2.7 は，線分をサイトとする場合の距離関数の例を示している．図にあるように点 p は，サイト s_i, s_j のボロノイ境界上の点であり，p から最も近い点は，それぞれ x_i, x_j となることを示している．

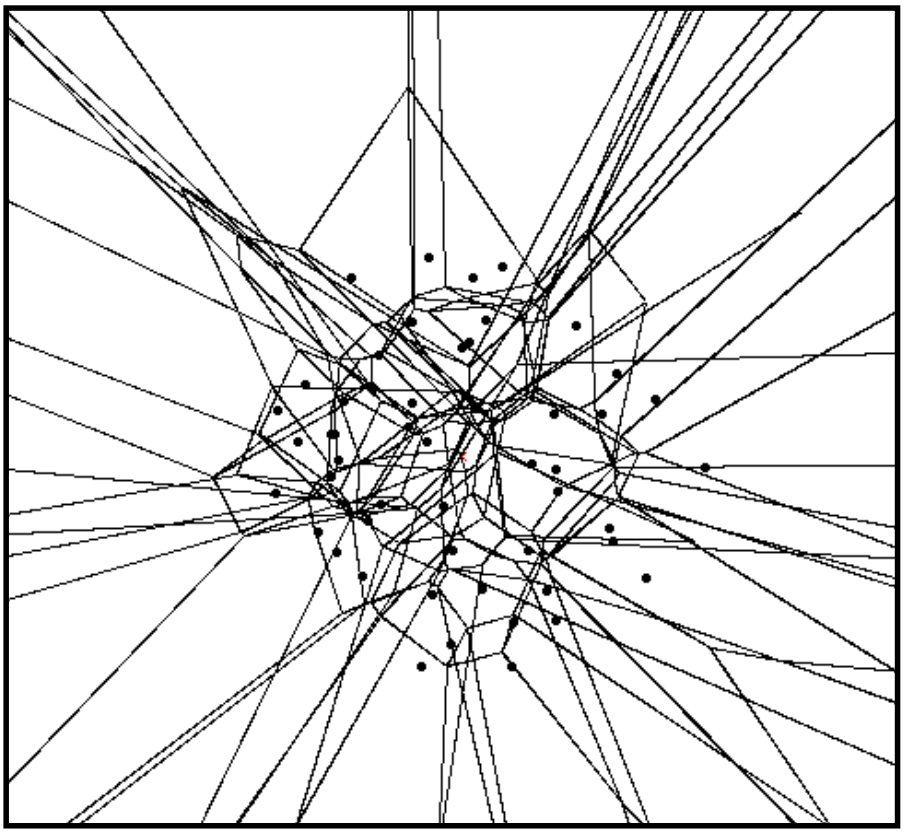

図 2.4: 3 次元ボロノイ図

サイトの形状を点に限らない場合，一般にボロノイ境界は直線だけでなく曲線からも構成されることになる．特に，図 2.7 に示すように線分のボロノイ図は，直線分および放物線で構成される．そのため，正確なボロノイ図を計算するためには，かなり精度の高い演算を必要とする．図 2.5, 2.6, 2.8 は，それぞれ線分のボロノイ図，円のボロノイ図，多角形のボロノイ図を示している．

2.3.3 距離関数における一般化

ある空間内の点とサイトの近さを表す際，これまでユークリッド距離を下で考えてきた．この距離関数を操作することによってボロノイ図は一般化することができる．ユークリッド距離の一般化形であるミンコフスキー距離 L_p は，次のように書ける．

$$d_{L_p}(p, p_i) = \left[\sum_{j=1}^{d} |x_j - x_{ij}|^p \right]^{1/p}$$

ただし，$p = (x_1, x_2, \ldots, x_d)$，$p_i = (x_{i1}, x_{i2}, \ldots, x_{id})$ とする．L_p の p は，点を表しているのでなく勢力の度合を表している．即ち，d_{L_2} はユークリッド距離そのものである．この

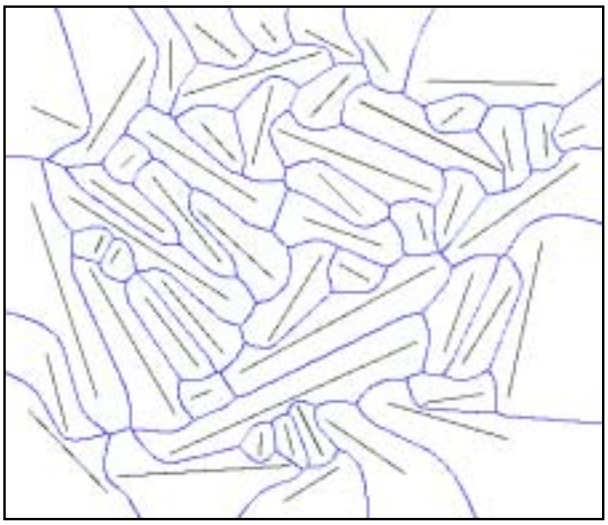

図 2.5: 線分のボロノイ図

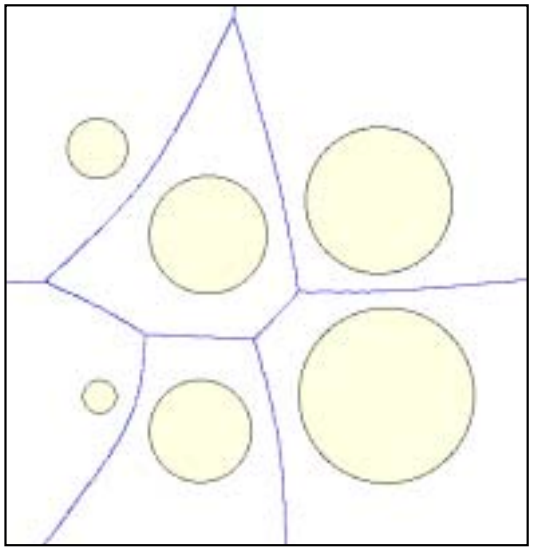

図 2.6: 円のボロノイ図

距離関数の下で得られるボロノイ図を L_p 距離のボロノイ図という．

また，与えられたサイトが個々に勢力を持つとき，このボロノイ図を重みつきボロノイ図と呼ぶ．このとき，あるサイト p_i の勢力が c_{i1}, c_{i2} によって表されるとき，

$$d_w(p, p_i) = \frac{1}{c_{i1}}\|\mathbf{x} - \mathbf{x}_i\| - c_{i2}$$

として定式化される．重みが大きいほど，ボロノイ領域は大きくなる．また，重み c_{i1} のようにユークリッド距離に対して掛けられたサイトがある場合，ボロノイ境界はアポロニウス曲線を有する．

2.3.4 その他の観点における一般化

近さの観点を変えて，全てのサイトから最も遠い点をそのサイトのボロノイ領域の点とするボロノイ図を最遠ボロノイ図（*farthest Voronoi diagram*）という．最遠ボロノイ図に対して，普通のボロノイ図を最近ボロノイ図と呼ぶこともある．最遠ボロノイ図は，空間を n 個の領域に分割するとは限らず，与えられたサイトの凸包上のサイトのみが領域を有するという性質を持つ．図 2.9 に最遠ボロノイ図の例を示す．同様の考え方から，k 番目に近い点に関するボロノイ図を総じて k 次のボロノイ図（*k-order Voronoi diagram*）と呼ぶ．

また，サイトが自由に移動することを想定し，逐次的にボロノイ図が更新されるものを動的なボロノイ図（*dynamic Voronoi diagram*）と呼ぶ．

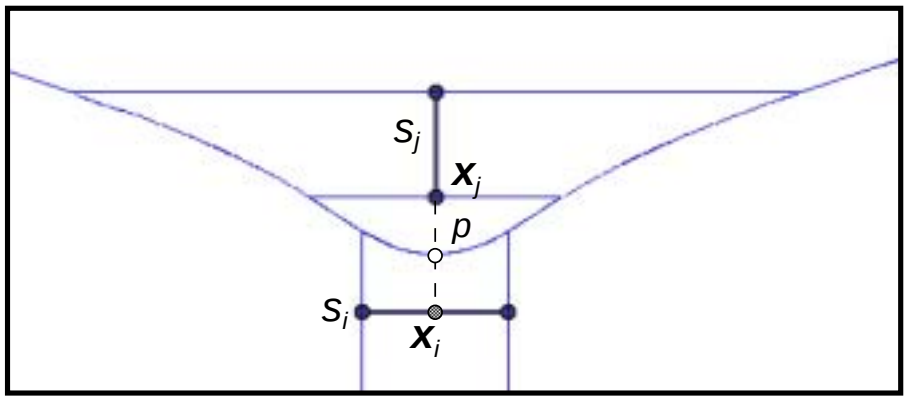

図 2.7: 線分の場合の距離関数

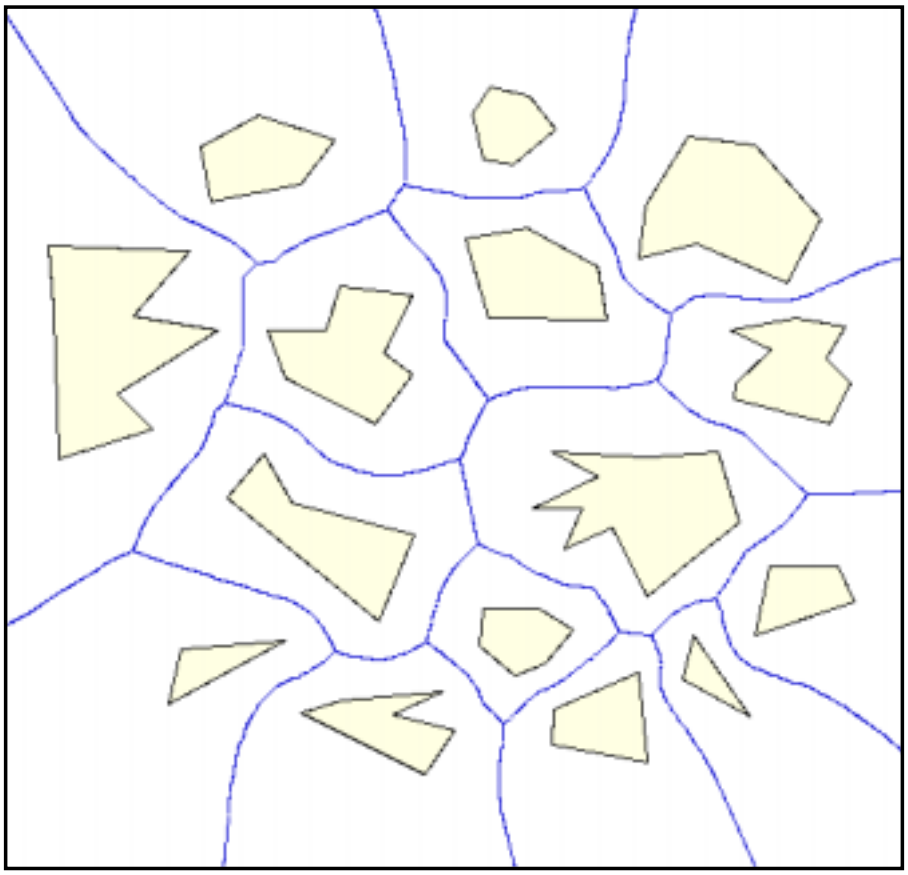

図 2.8: 多角形のボロノイ図

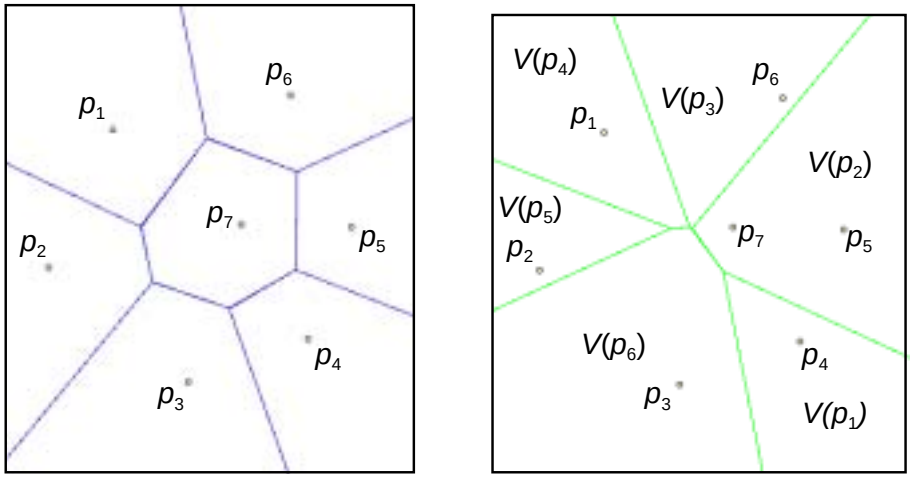

図 2.9: 最遠ボロノイ図と最近ボロノイ図

第3章　ハードウェア支援技法

ハードウェア支援技法は，グラフィックスアクセラレータのもつ特定の機能を利用するアプローチである．特に，ハードウェア支援技法でボロノイ図を求めるために，polygon rasterization, Z-buffer の機能を利用し，グラフィックスパイプラインによる高速化を実現する．本章では, ハードウェア支援によるボロノイ図の高速算法の基本的な原理, およびこの技法における誤差に関する設計上の注意点についてまとめる．

3.1　基本概念

ハードウェア支援技法によるボロノイ図の構成方法についてみていく．与えられたサイトに関して，その形状ごとに定義された 3 次元の距離関数を隠面除去の下で描画する．それをシーンに平行投影することによって得られた画像がボロノイ図となる．

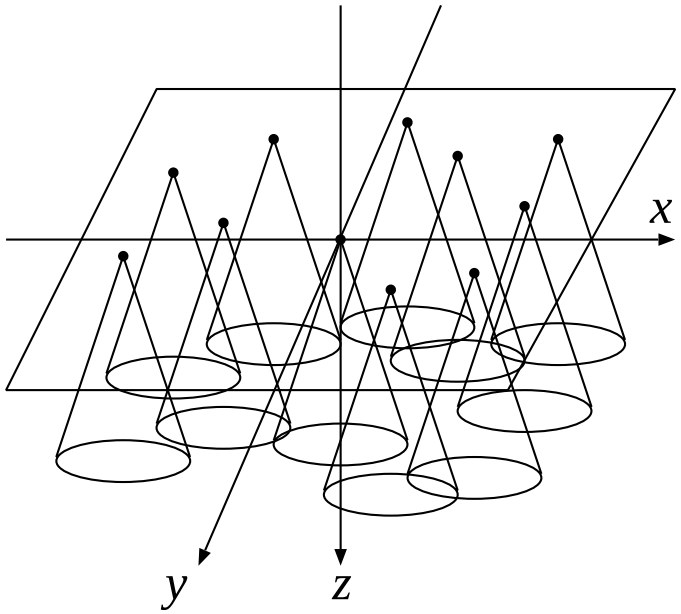

図 3.1: 点サイトのボロノイ図

図 3.1 に示すように，サイトの配置された xy 平面の垂直下向きに z 軸をとる．そして，各サイトを識別するためのカラー ID をもとに距離関数メッシュを描画していく．各画素

は自分の塗られた色によって，どのサイトに支配されているかが分かる．

3.2　隠面除去によるボロノイ領域の表現

depth バッファを利用するラスタ処理，即ち Z バッファ法（隠面除去）はハードウェア支援技法の根幹を担っている．Z バッファ法は，ポリゴンをレンダリングする際その z 座標値に従って，各ピクセルごとに出力するかどうかを判定する技法である．

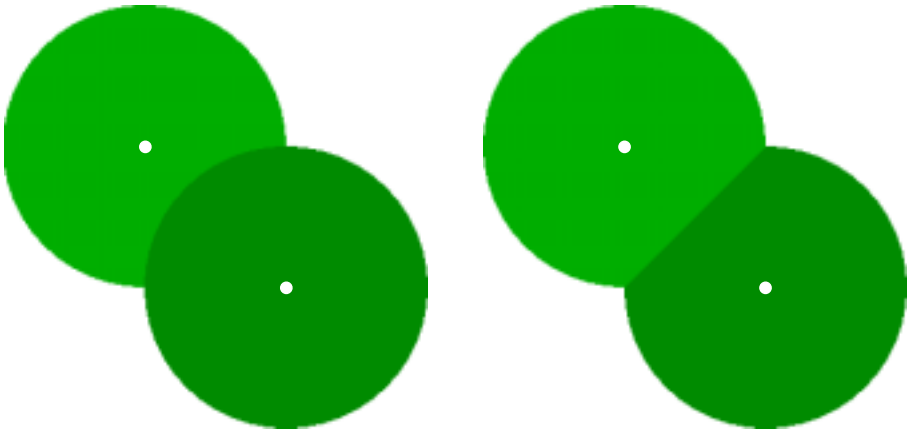

図 3.2: 隠面除去によるボロノイ領域の表現

図 3.2 に隠面除去による点ボロノイ図の境界の例を示す．同図（a）, (b）は 2 つの円錐を真上から見たもので，塗り重ね法による描画と隠面除去による描画の例をそれぞれ示している．特に，図 3.2（b）の色の変化している部分がボロノイ境界を表している．

3.3　各サイトの距離関数メッシュ

各サイトの形状（点，線分，多角形）によって，それぞれ距離関数メッシュが定義される．2 次元平面上では，点サイトは与えられた点を頂点とする直円錐となる．線分サイトは一方の端点から他方の端点まで直円錐を移動させてできる軌跡となる．すなわち，この軌跡はテントのような形状をとる 2 枚の長方形で表すことができる．また，多角形サイトに関しては点サイト，線分サイトの距離関数を組み合わせることで表現することができる．表 3.1 に，各サイトと距離関数メッシュの対応を示す．また，図 3.3 は実際の距離関数メッシュを示している．

表 3.1: 2D サイトの距離関数メッシュの形状

2D site	Shape of Distance Function
Point	直円錐　right circular cone
Line segment	”テント Tent”
Polygon	直円錐とテント

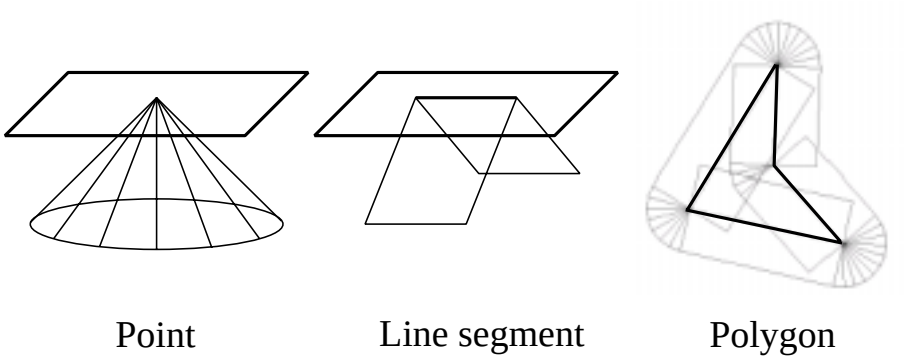

図 3.3: 各サイトの距離関数メッシュ

3.4　ハードウェア支援技法による誤差

3.4.1　距離関数近似による誤差

OpenGL などの一般的なグラフィックスライブラリは円錐や球などを直接描画することができないため，基本プリミティブ（三角形，もしくは四角形）で近似しなければならない．その近似により誤差が生じる．

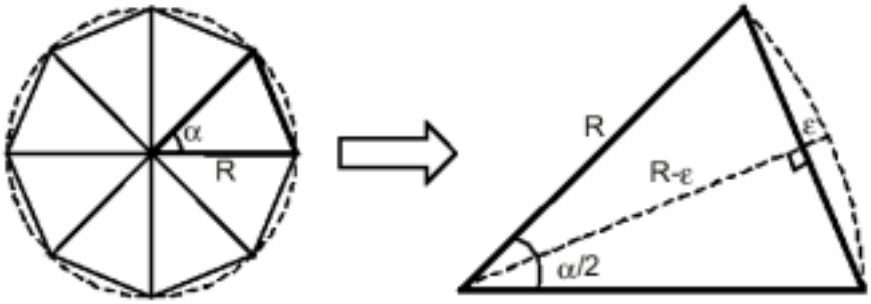

図 3.4: 円錐の基本プリミティブ近似による誤差

図 3.4 は，円錐を複数の三角形を用いて近似している例である．この誤差の最大値は ε である．ε が画素の対角線よりも大きい場合，誤った解を出力する可能性を持つ．この点

を考慮し，次式に基づき最適な近似円錐を得る．

$$\cos(\frac{\alpha}{2}) = \frac{R-\varepsilon}{R} \longrightarrow \alpha = 2\cos^{-1}\left(\frac{R-\varepsilon}{R}\right)$$

3.4.2 空間のサンプリングによる誤差

ハードウェア支援技法によって得られるボロノイ図はディジタル画像の形式であるため，ボロノイ図の精度は空間の点をどのようにサンプリングするかに依存する．即ち，解像度に依存することになるが，これはハードウェア支援技法によって得られる解の透過性から柔軟に対応することができる．画像の解像度における誤差は，狭いシーンに対してサイトを多数描画するような場合に起こりやすく，設計段階で十分注意しなければならない点である．

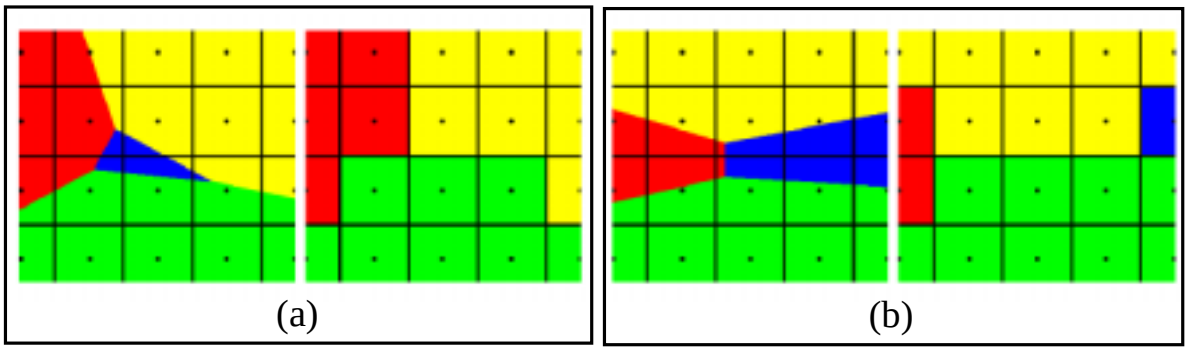

図 3.5: 誤差により位相が変化する例

誤差により位相が変化する例を図 3.5 に示す．図 3.5（a），(b）は，ボロノイ領域が丸められてなくなってしまう例と隣接関係に誤りを生じる例をそれぞれ示している．それぞれ，位相構造を保つために必要な部分にサンプリング点が無いことに起因していることが容易に確かめられる．単純な方法は，描画するシーン全体の大きさを大きくすることにより解像度を上げることで位相構造の歪みを無くすことができる．また，ディスプレイの大きさなどによりそれ以上解像度を上げることができないような場合は，誤差が発生していそうな部分を拡大・縮小することで対応することができる．

第4章　計算コストの比較

線分のボロノイ図を構成することにより，既存のソフトウェアパッケージ（LEP avd[4]）とハードウェア支援技法の実行時間に関する比較を行った．線分のボロノイ図を対象とした理由は，そのボロノイ境界が直線分および放物線から構成されるため，正しいボロノイ図を構成するためには厳密な計算を要するからである．このパッケージのアルゴリズムは，ランダム逐次添加法に基づいているため最悪の場合 $O(n^2)$，平均 $O(n)$ で求めることができる．ハードウェア支援技法でのボロノイ図の構成時間は，シーンの大きさ，解像度に依存する．今回は，500×500 のシーンでボロノイ図を計算した．また，avd ライブラリは線分どうしが交差するような入力に対しては計算することができないため，どの線分も交差しないような入力を生成して実験を行った．

実験は，サイト数 50, 100, 200, 500, 1000 に関しての実行時間の平均をとった．表 4.1 に実験を行った計算機環境を示す．実験環境は, 特別なグラフィックスハードウェアを搭載していない計算機上で行った．これは, どこにでもあるような普通の計算機であってもどれくらいの高速性が図られているのかをみたかったというのが理由である．図 4.1 には実験結果を示している．実験結果より，特別なグラフィックスハードウェアを使用していないにも関わらず，avd よりハードウェア支援技法の方がおよそ 10 倍程度処理効率に関して優れていることが確かめられた．また，ハードウェア支援技法では，点数が 500 辺りから計算時間が減衰していることが確かめられるが，これはシーンの大きさに対してサイト数が多いため解像度などの精度が不足しているということに起因していると考えられる．

表 4.1: 計算機環境

CUP	Pentium III 1.0GHz	OS	Linux 2.4.18
MEM	512MB	CC	gcc-2.96
VID	Intel 815, 16MB		

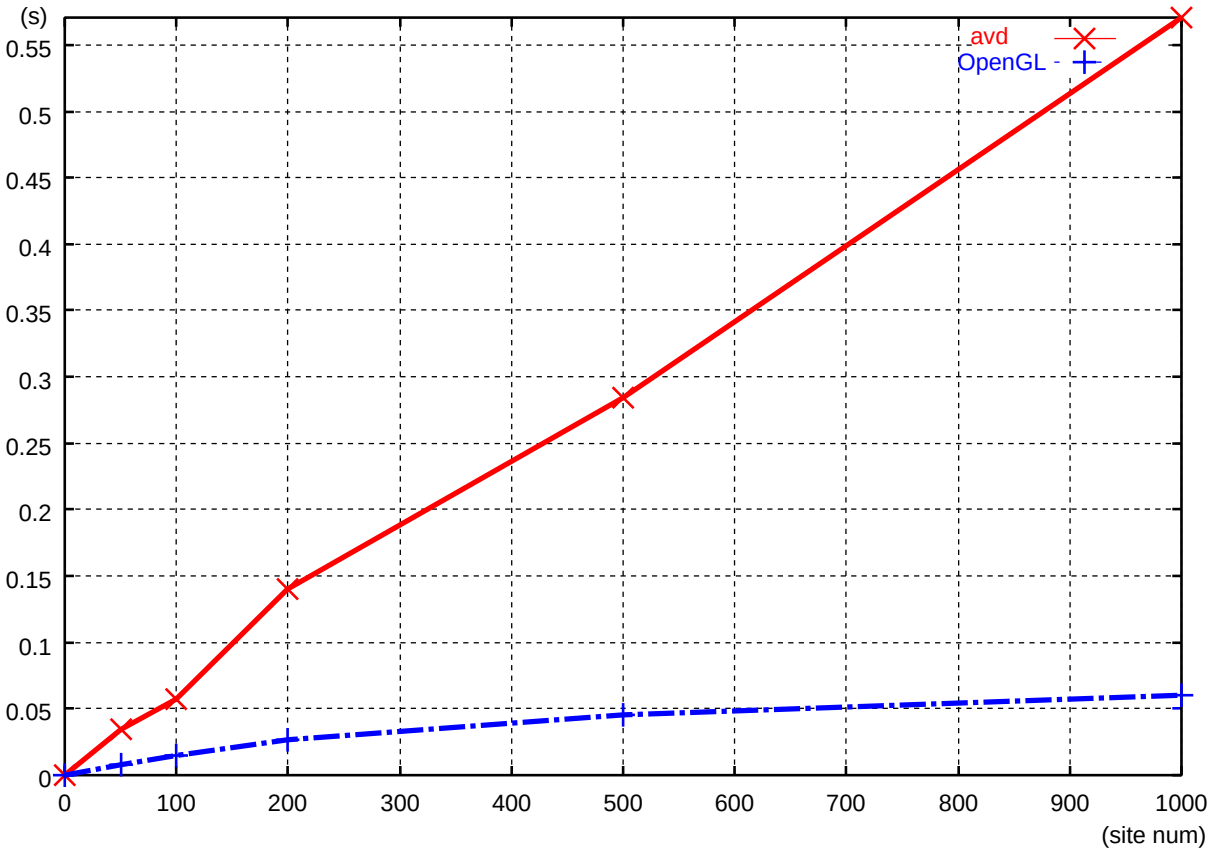
(s)
0.55
0.5
0.45
0.4
0.35
0.3
0.25
0.2
0.15
0.1
0.05
0
0
100
200
300
400
500
600
700
800
900
1000
(site num)
avd
OpenGL

図 4.1: 実行結果

第5章　いろいろなボロノイ図

一般化ボロノイ図は、ある応用問題に則して定義されたボロノイ図である．ボロノイ図の有益な性質を利用して多くの応用問題を解くために，様々な観点からその一般化がなされている．本章では，3 種類の一般化ボロノイ図をハードウェア支援技法で構成する方法を提案し，その応用問題について議論する．

5.1　L_1, L_∞ 距離でのボロノイ図

L_1 距離，L_∞ 距離ボロノイ図は，それぞれ次のような距離関数で定義される一般化ボロノイ図である．

$$\begin{aligned} d_{L_1}(p, p_i) &= \sum_{j=1}^{m} |x_j - x_{ij}| \\ d_{L_\infty}(p, p_i) &= \max_i \{|x_j - x_{ij}|, j = 1, 2, \cdots, n\} \end{aligned}$$

L_1, L_∞ 距離は，ユークリッド距離の一般形であるミンコフスキー距離に属する．ミンコフスキー距離は，次のように定義される．

$$d_{L_p}(p, p_i) = \left[\sum_{j=1}^{m} |x_j - x_{ij}|^p \right]^{1/p} \tag{5.1}$$

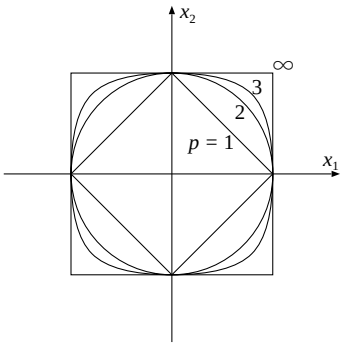

図 5.1: 1,2,3,∞ に関するミンコフスキー距離の等距離線

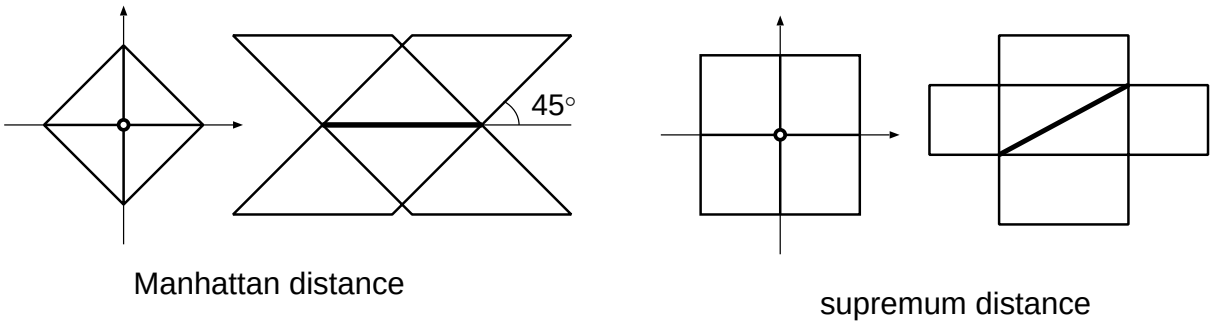

図 5.2: L_1, L_∞ 距離の距離関数メッシュ

図 5.1 は，式 5.1 のミンコフスキー距離において $p = 1, 2, 3, \infty$ の場合の等距離線を示している．この図は，L_1, L_∞ 距離のボロノイ図をハードウェア支援技法で構成するためのアイディアを与えている．即ち，L_1, L_∞ 距離の距離関数メッシュは与えられた点サイトを頂点とする四角錐によって表現される．互いに同形であり，頂点に関して距離関数を 45 度回転させることで形状は一致する．図 5.2 に L_1, L_∞ 距離の距離関数メッシュの形状を示す．

図 5.3 は，ハードウェア支援技法による L_1, L_∞ 距離のボロノイ図の例をそれぞれ示している．左の画像が L_1 距離のボロノイ図，右の画像が L_∞ 距離のボロノイ図であり，ともに同じ点集合を入力としている．

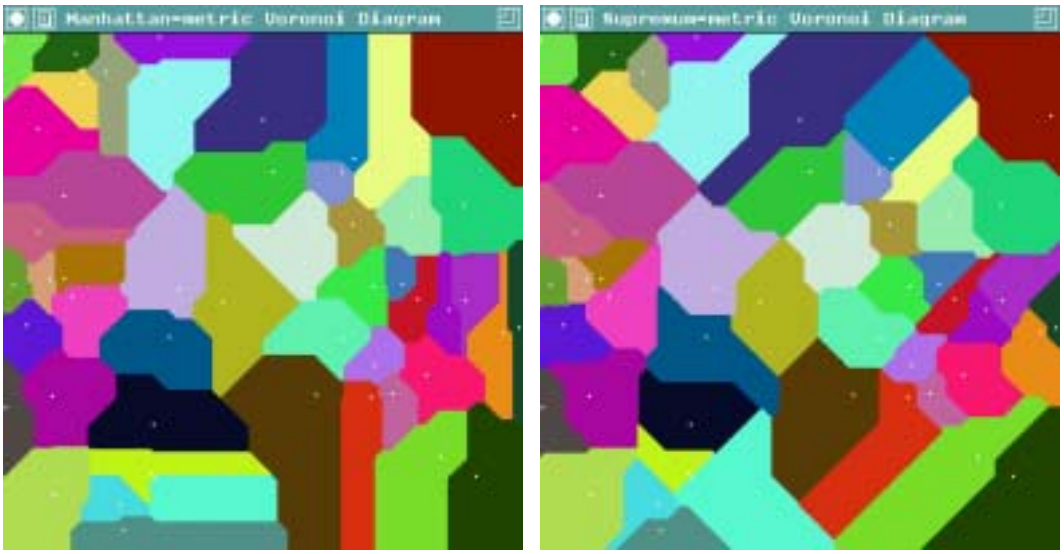

図 5.3: L_1, L_∞ 距離のボロノイ図の出力例

L_1, L_∞ 距離は，それぞれマンハッタン距離 ($Manhattandistance$)，上限距離 ($supremum\ distance$) と呼ばれ，移動方向に制限がある空間における距離を表すために用いられている．

計算幾何学におけるパラダイムとして分割統治法があり，$O(n \log n)$ の計算時間と $O(n)$ 記憶量のアルゴリズムが知られている [6]．サイト形状に関する一般化に関しては，あまり行われていない．しかし，線分や多角形集合に関する L_1, L_∞ 距離のボロノイ図は，ハードウェア支援技法では容易に実装することができる．

[6] では，応用問題としてこのメトリック空間における最短ハミルトンパス問題の近似解法について扱っている．また，その他の応用例として回路基盤の配線や道路地図の簡略図示などがあり，ハードウェア支援技法による効率化が期待される．

5.2 楕円距離のボロノイ図

平面上の 2 点 $p = (x_1, y_1), q = (x_2, y_2)$ に関して，楕円距離を以下のように定義する．

$$d_{ellip}(p, q) = \sqrt{a(x_1 - x_2)^2 + 2b(x_1 - x_2)(y_1 - y_2) + c(x_2 - y_2)^2}$$

ただし，a, b, c は $ac > b^2$ を満たす実数とする．この距離関数 d_{ellip} に基づくボロノイ図を楕円距離のボロノイ図という．図 5.4 に，楕円距離のボロノイ図の例を示す．普通のボロノイ図のボロノイ頂点は，3 点以上のサイトを通る円の中心であったのに対し，楕円距離のボロノイ図のボロノイ頂点は楕円の中心となる．

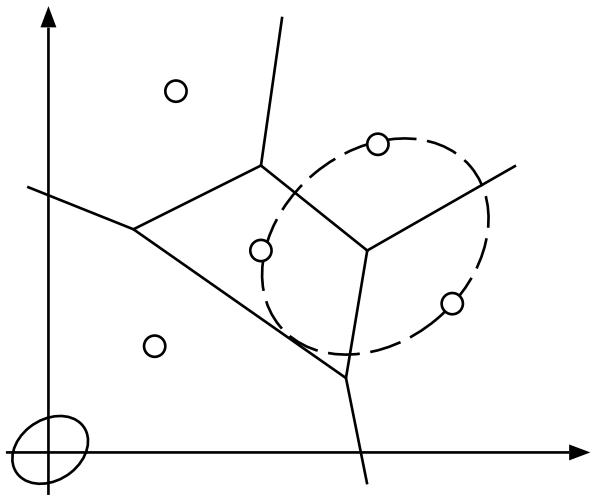

図 5.4: 楕円距離のボロノイ図

楕円距離のボロノイ図は凸距離のボロノイ図の一種であり，次のような手続きで得ることができる．まず，最初に与えられた点集合およびその平面全体に対してアフィン変換を施す．次に，アフィン変換された点集合に対してユークリッド距離でのボロノイ図を求める．最後に，得られた図形を先ほどの逆変換を行い元の平面へ移すことで楕円距離のボロノイ図を求めることができる．

ハードウェア支援技法では，まず最初に普通にボロノイ図を描画し，その画像全体に対して変換行列を掛けることで得られる．

$$M = \begin{pmatrix} 1 & \cot\theta & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \tag{5.2}$$

例えば，式 5.2 のようなせん断を行う行列を用いてアフィン変換を行うと図 5.5 のように楕円距離のボロノイ図を得ることができる．左の図が，変換行列を掛ける前のボロノイ図で，右の図画変換行列を掛けた楕円距離のボロノイ図を示している．

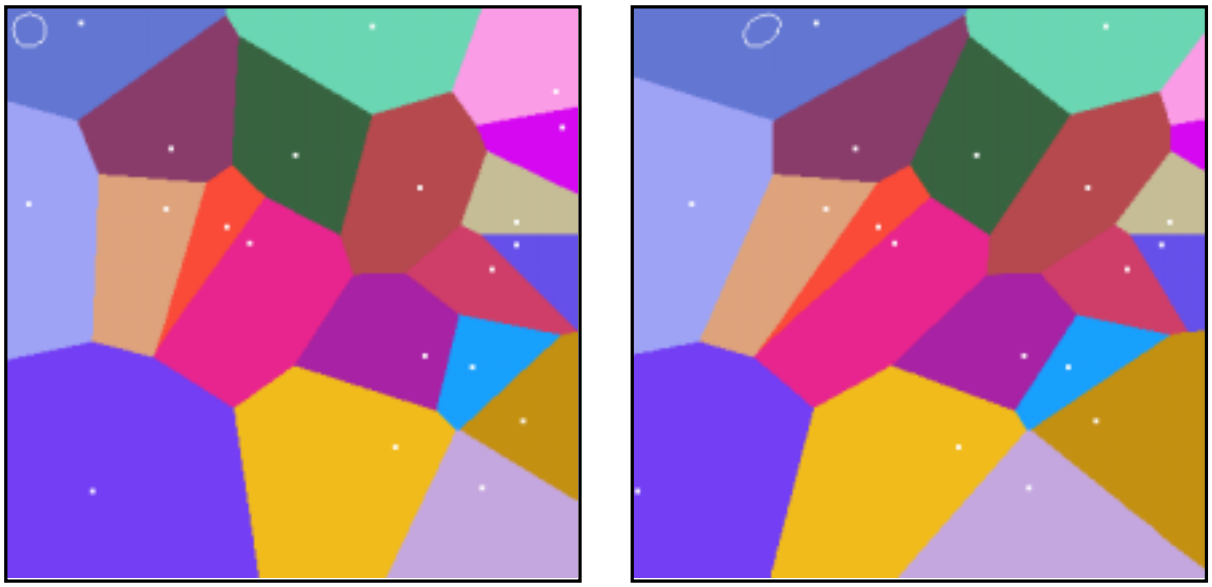

図 5.5: 楕円距離のボロノイ図

楕円距離は，移動する方向によって移動コストが異なるような空間を表すために用いられる．即ち，より幾何学的な制約のある空間を想定した場合といえる．真に平坦な平面の距離は，ユークリッド距離によってその移動コストを測ることができるが，部分的に線形な面を持つ平面（真に平坦でない平面）の場合その移動コストをどのように扱うかという問題が起こる．この場合，平面をタイルのような四辺形によって分割し，各タイル固有の楕円距離を想定することによって移動コストを測るというアプローチがとられている．真に平坦でない平面におけるボロノイ図は，各タイルごとに部分的なボロノイ図を求め，それを繋ぎ合わせることで求めることができる．

5.3 加法，乗法，複合重みのボロノイ図

加法重み付きボロノイ図，乗法重み付きボロノイ図，および複合重みのボロノイ図は，それぞれ以下に示す距離関数の下で与えられる一般化ボロノイ図である．

$$
\begin{aligned}
d_a(p, p_i) &= ||p - p_i|| - w_i \\
d_m(p, p_i) &= \frac{1}{w_i}||p - p_i|| \\
d_c(p, p_i) &= \frac{1}{w_{i_1}}||p - p_i|| - w_{i_2}
\end{aligned}
$$

本節では，加法重み付きボロノイ図における距離関数を定式化する．乗法重み付ボロノイ図については，第 6 章で取り扱う．複合重み付きボロノイ図は，本節および第 6 章の重み付けの方法を組み合わせることで実現できるので割愛する．

加法重み付き距離において，あるサイト p_i の重みが w_i であるとき，サイト p_i は距離関数の定義より中心が p_i で半径が w_i の円とみなすことができる．サイト集合 $P = \{p_1, \ldots, p_n\}$ および重み集合 $W = \{w_1, \ldots, w_n\}$ に関して，中心 p_i，半径 w_i とする円を $c_i(p_i, w_i)$ と表し，その集合を $C = \{c_1, \ldots, c_n\}$ とする．このとき加法重み付きボロノイ図は C に関する円のボロノイ図に等しい．

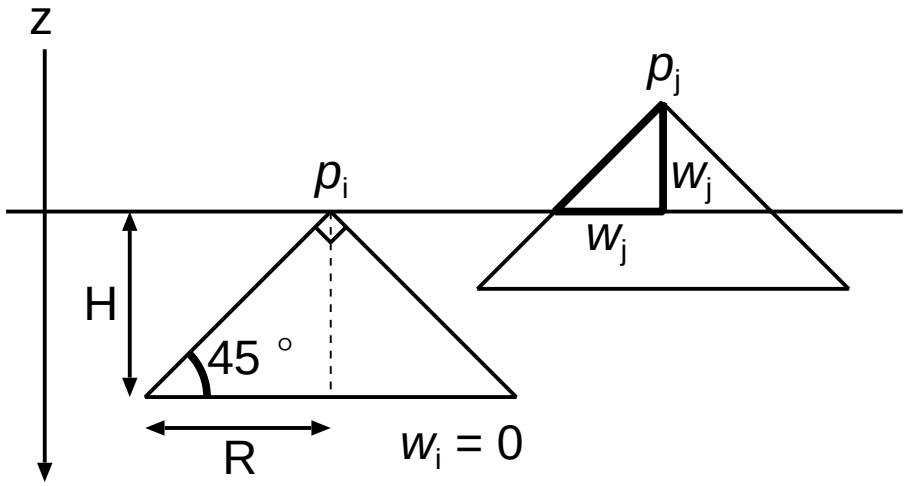

図 5.6: 加法重み付きボロノイ図に対する 3 次元距離関数

図 5.6 は，加法重み付きボロノイ図に対する 3 次元距離関数を示している．3 次元距離関数としては，円錐の頂上の角が直角のものとする．このときサイト s_i の重み w_i を表現するには，w_i だけ z 軸の負の方向に平行移動させることでできる．つまり，(x, y) の距離関数メッシュの頂上は，$(x, y, -w_i)$ となる．これで，距離関数とシーンの共通部分は，半径 w_i の円になる．図 5.7(a)(b)(c) は，それぞれ同じ点集合に対する普通のボロノイ図，乗

法重み付きボロノイ図，加法重み付きボロノイ図を示している．加法重み付きボロノイ図の場合，各サイトは必ずしも領域を持つとは限らないことに注意されたい．

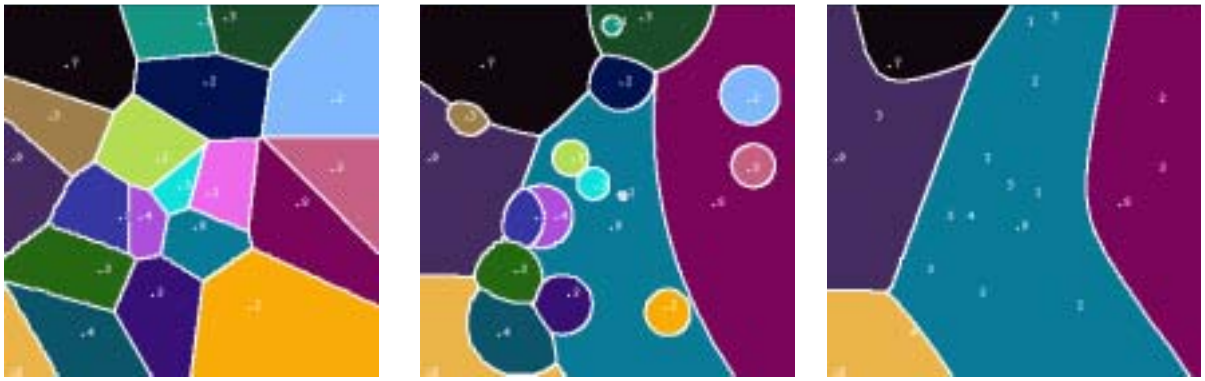

図 5.7: ハードウェア支援技法による各種重み付きボロノイ図の例

重み付き距離は, 各サイトがそれぞれ異なる勢力を持つような場合を表現するために用いられる．例えば，価格や品揃えなど顧客のニーズを反映するよう重み付けがなされた商圏を考えたとき，どのサイト (商店) に顧客が流れやすいかを可視化するような場合に重み付きボロノイ図が利用される．また，乗法重み付きボロノイ図の応用として第 6 章で等高線地図の補間アルゴリズムを提案する．

5.4 Hausdorff 距離のボロノイ図

Hausdorff 距離は，集合間の距離を測る尺度を与える．平面上の任意の点 p から有限サイト集合 A_i への距離を次のように定義する．

$$d_H(p, p_{ij}) = \max_{p_{ij}}\{||p - p_{ij}|| \ |p_{ij} \in A_i\}$$

距離関数 d_H での $A = \{A_1, \cdots, A_n\}$ に関するボロノイ図を Hausudorff 距離のボロノイ図という．この距離関数でのボロノイ図は，点集合の凸包をサイトとする最遠点ボロノイ図となる．ハードウェア支援技法では，凸包を計算せずに各集合ごとにカラー ＩＤを割り当てることで凸包の境界内のサイトに関しても描画してしまうことにする．図 5.8 に，ハードウェア支援技法による Hausdorff 距離のボロノイ図を示す．

Hausdorff 距離は，ある集合間の違い度の計算に用いられる．この集合間の違い度をどのように決めるか，ということはパタン認識やコンピュータビジョンにおいて重要なテーマのひとつである．[17] は，より一般的な Hausdorff 距離のボロノイ図を扱っている．入力として与えられる 2 つの図形 (集合) に対して，平行移動，回転を許した場合の Hausdorff

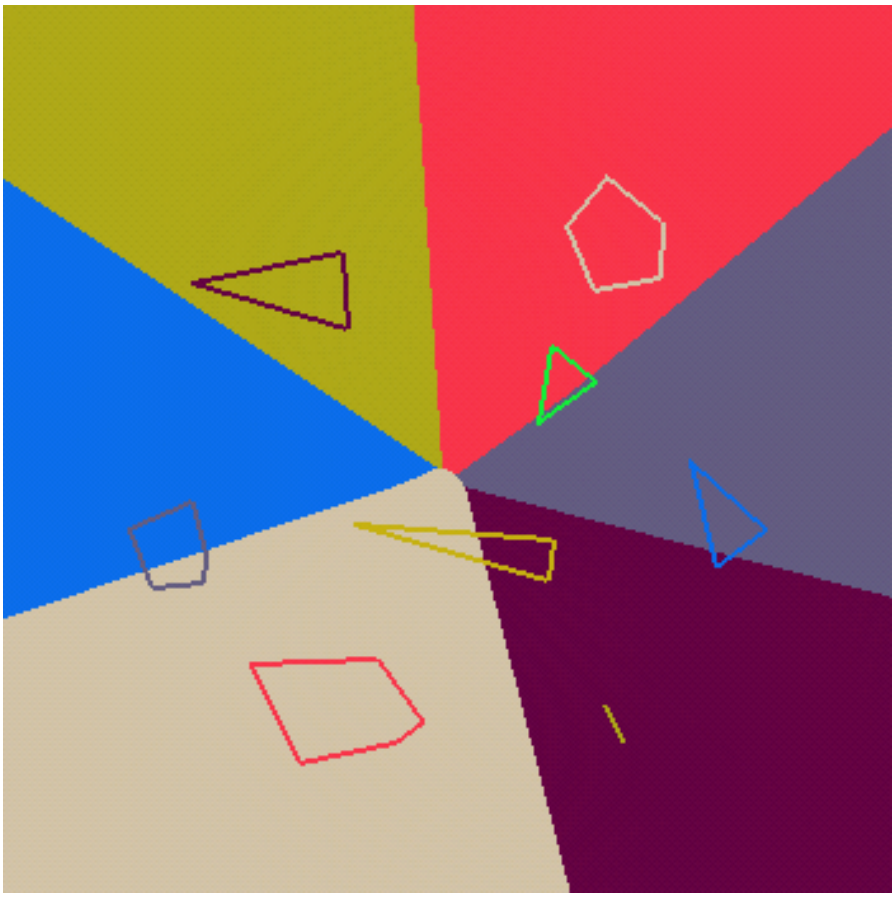

図 5.8: Hausdorff 距離のボロノイ図の例

距離の最小値を違い度とし，この違い度の計算にボロノイ図の性質を利用している．この文献によると，最小 Hausdorff 距離の計算には $O((m+n)^6 \log(mn))$ 時間かかり，実際の応用問題に適用するにはあまり実用的でないといえる．一方で，ハードウェア支援技法では，図形 (集合の点) に関する平行移動，回転はグラフィックスパイプラインの並行処理を利用できるため，計算時間の効率化が期待できると考えられる．

5.5 その他

本論文では取り上げなかったがその他にも凸距離のボロノイ図，ボート・オン・リバーボロノイ図，動的なサイトのボロノイ図など，多彩な一般化ボロノイ図も構成できるであろうと考えられる．

第6章　等高線地図の補間

等高線地図の補間は，データの無い地域の標高を推定し補助等高線を付加することである．一般に，精密的方法と近似的方法に区分されるが，後者のデータを平滑する方法に基づきハードウェア支援技法を適用する．各等高線をサイトとするボロノイ図の境界はその補助等高線となる．また，重みの付けられた距離関数による一般化ボロノイ図を構成することにより，補助等高線を高速に得るためのアルゴリズムを提案する．

6.1　乗法重み付きボロノイ図

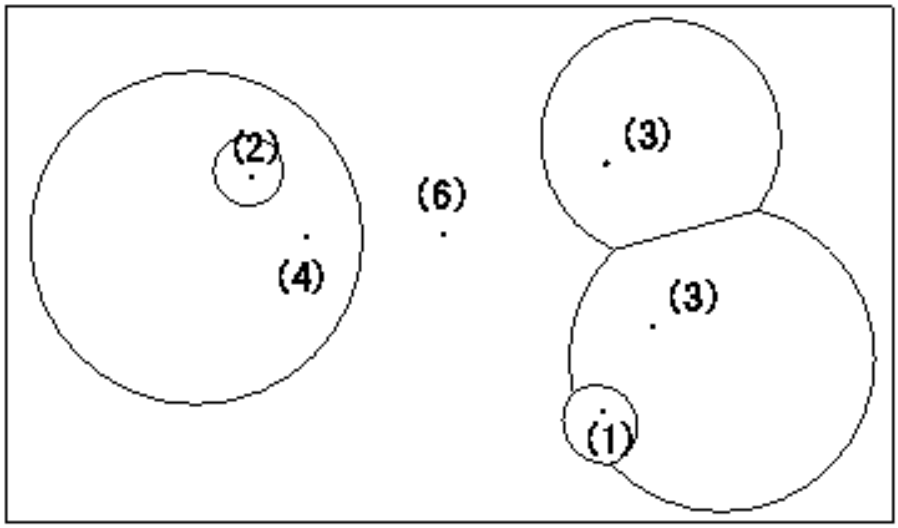

図 6.1: 乗法重み付きボロノイ図

式（2.5）において，距離関数 $dist$ を次式で定義したボロノイ図を乗法重み付きボロノイ図という（図6.1）．ただし，w_i はサイト s_i が持つ荷重とする．このボロノイ図は，任意の比率のボロノイ領域を形成することができる．

$$dist(\mathbf{p}, s_i) = \frac{1}{w_i}||\mathbf{p} - s_i||$$

6.2　ハードウェア支援技法による実現

各サイトへの重み付けは距離関数メッシュの高さをパラメータとし，式（6.1）のように表現することによって実現することができる．

$$h_i = \frac{H}{w_i} \tag{6.1}$$

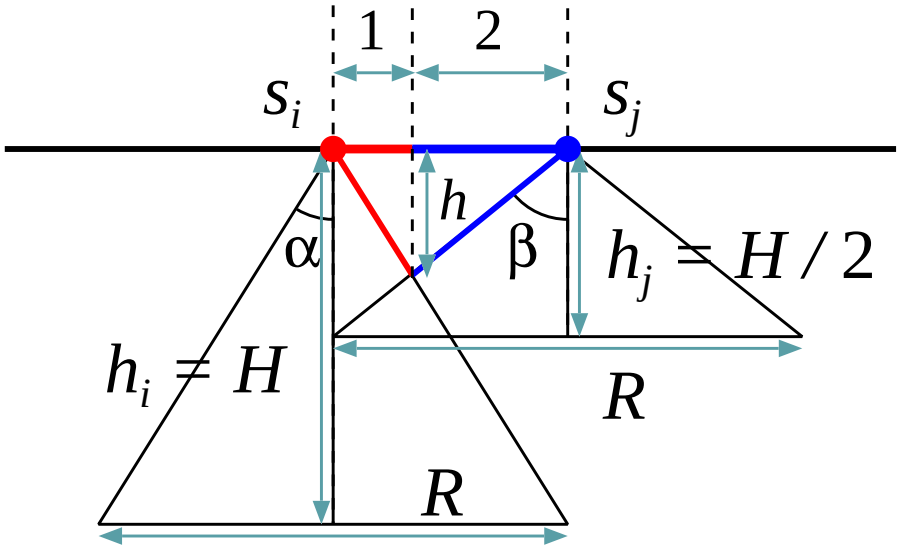

図 6.2: 重み比が 1 : 2 の場合

ただし，h_i はサイト s_i の距離関数メッシュの高さとし，H をデフォルトの高さとする．図 6.2 の重み比が 1：2 の場合を例にとると，その正しさは次のように示すことができる．

$$\tan\alpha = \tfrac{1}{h},\quad \tan\beta = \tfrac{2}{h} \quad \Rightarrow \quad \tan\alpha = 2\tan\beta$$
$$\tan\alpha = \tfrac{R}{2h_i},\quad \tan\beta = \tfrac{R}{2h_j}$$
$$2\cdot\tfrac{R}{2h_i} = \tfrac{R}{2h_j} \quad \Rightarrow \quad 2h_j = h_i$$

以上より，アルゴリズムを疑似コードで示すと以下のようになる．

入力：等高線集合 $C\{c_1, c_2, \ldots, c_n\}$，補間率 $\alpha(\geq 3)$.

出力：補助等高線集合 $C'\{c'_1, c'_2, \ldots, c'_m\}$.

Contour Interpolation

```
1  C' = C
2  for (i = 1; i < α; i++) {
3      for (j = 2; j ≤ |C|; j++)
4          c_j の距離メッシュの高さを (α−i)h_{j−1} / i とする.
5      ボロノイ図を構成
```

6　　ボロノイ境界を検出し，C' に挿入
7　}
8　return C'

6.3　実行結果

図 6.3 に，等高線地図の補間アルゴリズムの実行結果の例を示す．左の図は，入力等高線およびそのボロノイ図を示しており，右の図は，入力等高線に対して補間率 α （等高線間に $\alpha - 1$ 本の補助等高線を引く）での実行結果を示している．

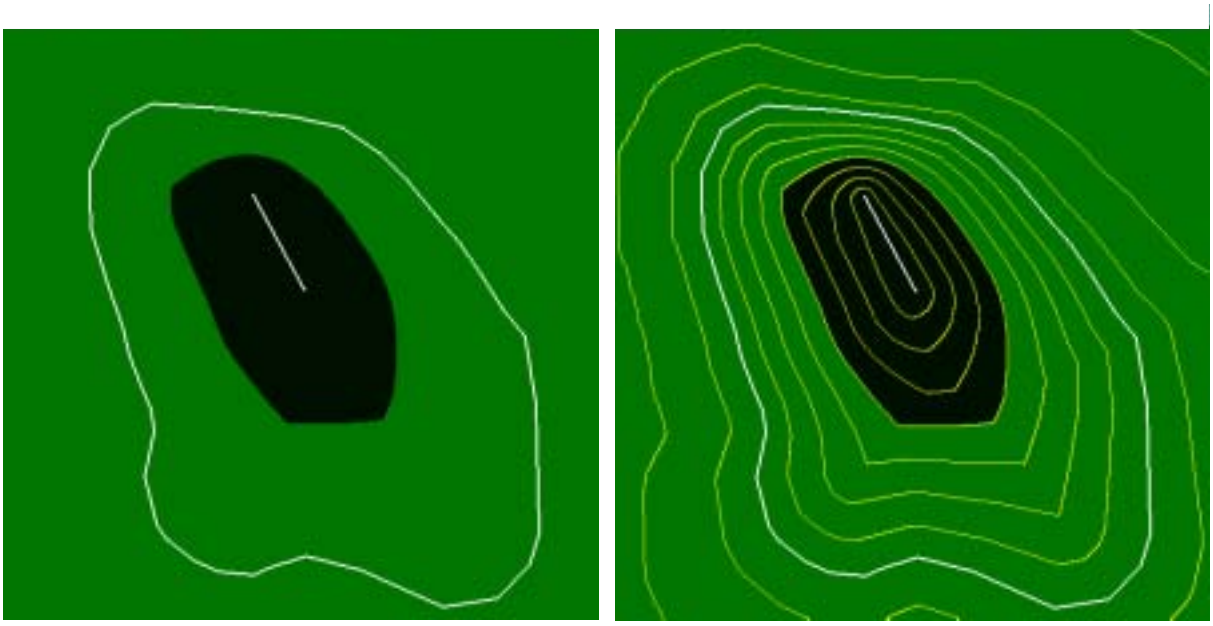

図 6.3: 等高線地図の補間の例

6.4　提案アルゴリズムの漸近的解析

本節では，上記の提案アルゴリズムの計算時間に関する漸近的解析を示す．与えられる等高線集合の要素数を n とする．また，入力としていされる補間率を $\alpha \geq 1$ と表記する．各サイトへの重み付けは，それぞれ定数時間で実行できる．ボロノイ図の描画時間を t_1，ボロノイ図の検出にかかる時間を t_2 と置くと，アルゴリズムの計算時間を $T(n)$ は，$T(n) = (\alpha - 1)(n + t_1 + t_2)$ と書ける．ボロノイ図を描画する時間 t_1 は，確にサイト数に依存するがグラフィックスハードウェアにより高速に処理されるため，$o(n)$ として考えることができる．ボロノイ図の検出は，乗法重み付きボロノイ図を描画しているため，非連結

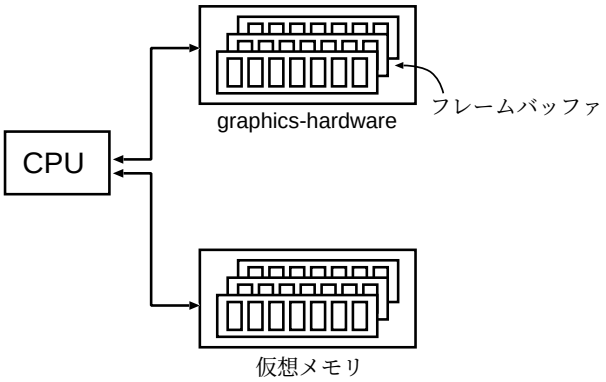

図 6.4: グラフィックスハードウェアと仮想メモリ

な境界ができてしまい境界を辿るような出力サイズに敏感なアプローチはできない．従って，シーン全体を見て回らなければならないため，アルゴリズムのボトルネックとなる．

この境界抽出におけるボトルネックは，グラフィックスハードウェア内のフレームバッファから CPU を介して仮想メモリ内に画像を取り込まなければ処理ができないということに起因している．図 6.4 に，CPU，グラフィックスハードウェア，および仮想メモリの基本的な構造を図示する．上記のことを踏まえ, ボロノイ境界の検出をグラフィックスハードウェア内のフレームバッファ上ですべて対処できるような手法が要求される．この問題に関しては，今後の課題として残しておく．

第7章　今後の課題

今回提案した補助等高線を求めるアルゴリズムは，まだ仮想的な等高線地図でしか実験していない．今後の課題として実際の等高線地図においてこれを適用し，得られる補助等高線に関して考察する．

また，この等高線補間のアルゴリズムを発展させ，画像の等高線表現による階調補間について考察する．画像の等高線表現は，各ピクセルの RGB 値を標高とし，画像全体を一つの等高線地図とみなす表現方法である．この表現は加工の際，画像の大域的な特徴を保持することができ，画像のキズ除去，解像度補填，そして自然な拡大縮小などの処理に優れている．図 7.1 に，画像の等高線表現の例を示す．画像の等高線表現に対して，等高線

320	320	330	330	330	330
310	320	330	340	340	330
310	310	320	330	340	330
310	310	320	320	330	330
300	310	310	320	320	320
300	300	310	310	320	320

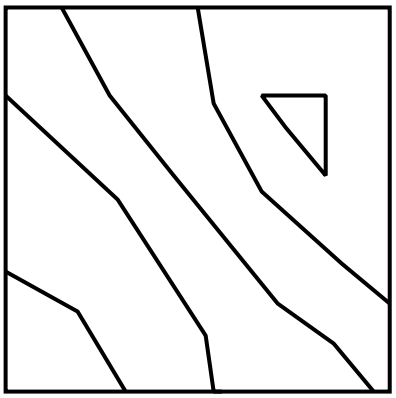

図 7.1: 画像の行列表現と等高線表現

地図の補間のアルゴリズムを適用することにより，高速に多階調の画像が得られるのではないかと考えている．

謝辞

研究を進めるにあたって多大な御指導ならびに御支援頂いた浅野哲夫先生に心から感謝します．中野浩嗣先生には，実験環境ならびに貴重な意見を頂き感謝します．浅野研究室をはじめとする情報基礎学講座の皆さんには，ゼミなどでのディスカッションを通して多くの貴重な御意見を頂き感謝します．特に，小保方幸次助手，本学大学院生の河村泰之さんにはプログラミングの面で多大な御指導を頂き感謝します．池田隆之氏，千葉英史氏には研究に対する多くの討論を承り感謝します．最後に，生活面で援助をしてくれた家族に感謝します．

参考文献

[1] K.E.Hoff III, T.Culver, J.Keyser, M.Lin, D.Manocha: Fast Computation of Generalized Voronoi Diagrams Using Graphics Hardware, *Proc. ACM SIGGRAPH*, pp.277-285, 1999.

[2] N.Mustafa, E.Koutsofios, S.Krishnan, Suresh V.: Hardware-Assisted View-Dependent Map Simplification, Comp. Geom(SOCG '01), pp. 50-59, 2001.

[3] 杉原厚吉, 藤村光, 山本修身: 勢力圏図の高速計算とそれを利用したスポーツチームワークの解析, 公開シンポジウム「アルゴリズム工学」講演予稿集, pp. 43-52, 2001.

[4] Michael Seel: LEP Abstract Voronoi Diagrams, http://www.mpi-sb.mpg.de/LEDA/friends/avd.html.

[5] A.Okabe, B.Boots, and K.Sugihara: Spatial Tessellations -second edition-, John Wiley & Sons, Chichester, UK, 1999.

[6] D.T.Lee, C.K. Wong: Voronoi Diagrams in $L_1(L_\infty)$ Metrics with 2-Dimensional Storage Applications, SIAM J. COMPUT. Vol.9, No.1, 1980.

[7] K.Sugihara, M.Iri, H.Inagaki, T.Imai: Topology-Oriented Implementation–An Approach to Robust Geometric Algorithms, Algorithmica 27(1), pp. 5-20, 2000.

[8] M.Held: VRONI: An engineering approach to the reliable and efficient computation of Voronoi diagrams of points and line segments, Computational Geometyr 18, pp. 95-123, 2001.

[9] S.Fortune: A Sweepline Algorithm for Voronoi Diagrams, Algorithmica vol.2, pp. 153-174, 1987.

[10] K.Q.Brown: Voronoi Diagrams form convex hulls, Information Processing Letters vol.9, pp. 223-228, 1979.

[11] H.Edelsbrunner, R.Seidel: Voronoi diagrams and arrangements, *Discrete and Computational Geometry*, vol.1, pp.25-44, 1986.

[12] T.Ohya, M.Iri, K.Murota: Improvement 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, 1984.

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

[14] C.Burnikel, R.Fleischer, K.Mehlhorn, S.Schirra: Efficient exact geometric computation made easy, in: Proc. 15th Annu. ACM Symos. Comput. Geom., Miami Beach, FL, June 1999, pp. 341-350.

[15] K.Mehlhorn, S.Naher: LEDA. A Platform for Combinatorial and Geometric Computing, Cambridge University Press, 1998.

[16] V.Karamcheti, C.Li, I.Pechtchanski, C.Yap: A core library for robust numeric and geometric computation, in: Proc. 15th Annu. ACM Sympos. Comput. Geom., Miami Beach, FL, June 1999, pp. 351-359.

[17] D.P.Huttenlocher, K.Kedem, J.M.Kleinberg: On Dynamic Voronoi Diagrams and the Minimum Hausdorff Distance for Point Sets Under Euclidean Motion in the Plane, 8th Annu. ACM Sympos. Comput. Geom., 1992, pp. 110-119.