ボロノイ図を用いた画像のステンドグラス化手法

芳賀俊之　　西田友是

東京大学大学院 理学系研究科 情報科学専攻

E-mail：{haga, nis}@is.s.u-tokyo.ac.jp

市販の画像編集ソフトには、画像に対して各種の効果を付加する機能があり、“ ステンドグラス化 ”もその一つである。しかし、その結果は十分満足できるものではない。そこで、ボロノイ図を用いてステンドグラスの各領域を表現し、自動的にステンドグラス風画像の生成を行う手法を提案する。提案法は、大きく二つのプロセスから成り、1) 単純な操作の組み合わせにより、入力画像の特徴 (輪郭等) を表現した、ボロノイ領域の生成を行うモデリングプロセスと、2) 得られた母点やボロノイ辺の情報から、よりステンドグラス風な画像を生成するレンダリングプロセスである。また、グラフィックスハードウェアを利用することで、効率的な画像生成を可能にした。

A Rendering Method of Stained-Glass-Style Images Using Voronoi Diagrams

Toshiyuki Haga　　Tomoyuki Nishita

Department of Information Science, Graduate School of Science,
The University of Tokyo

Commercial image editing softwares provide many functions to add various effects to images. One of the functions is to convert images into stained-glass-style images. However, the resulted images are not satisfiable. Therefore we present a method to generate stained-glass-style images automatically, expressing each region of stained glass by Voronoi diagrams. The proposed method consists of two main processes; one is the modeling process that generates the Voronoi regions taking into account the features (e.g. boundaries) of the input image by performing a combination of several simple operations, and another is the rendering process that creates the image by using the Voronoi generators and Voronoi edges calculated, in order to make the image to be similar to stained glass. Using graphics hardware, we can generate stained-glass-style images efficiently.

1　はじめに

コンピュータの普及に伴い、専門でない人でも CG に触れ、作成する機会が増しており、フォトレタッチソフト等も頻繁に利用されている。市販のソフトウェアには、入力画像に対して各種の効果を付加する機能が備わっており、すばやく、簡単に、様々な画像を作ることができる。しかし、それらの機能は、(時間的にも空間的にも) 低コストで行うことを前提としており、必ずしも満足のいく結果を得ることはできない。

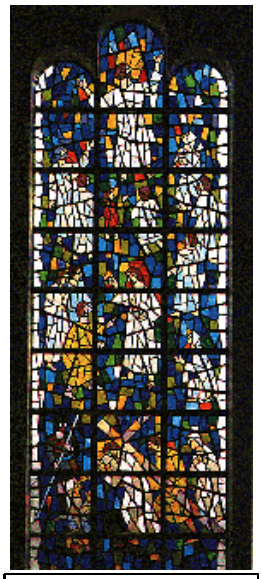
その一つに、画像のステンドグラス化というものがある。その例として、図 1 に市販のソフトウェアによる結果画像を示すが、それには、元画像の特徴が良く表現されているとは言い難い。多少時間を要してでも、より品質の高い結果画像が望まれることもあるだろう。

図 1: 元画像 (左) と“ ステンドグラス化 ”した画像 (右)

また、近年、コンピュータ関連機器は低価格化が進んでおり、安価に高性能なものを手に入れられる。グラフィックスハードウェアもその一つであり、手軽に高品質な画像・映像を楽しめる一つの要因となっている。

ここで、実際のステンドグラス (図 2) を見てみると、ガラスとガラスの境界により、輪郭線等の特徴を表し、各ガラス領域は概ね凸多角形で構成されている。そこで、これらをボロノイ図により近似し、画像を生成することを考える。このような近似によって、ハードウェア化も容易になり、高速化が望める。

以上の点を踏まえ、ユーザーが容易に制御することができ、かつ、より高品質なステンドグラス風画像を生成可能である、グラフィックスハードウェアを利用した手法を提案する。



東北学院大学 泉キャンパス 礼拝堂ステンドグラス[1]

図 2: 実際のステンドグラスの例

2 関連研究

ハードウェアを利用したボロノイ図生成

画像精度のボロノイ図[2] をハードウェアを用いて高速に生成する方法は、プログラミングガイド [1] にも記述がある。その方法を提案法でも採用しているので、簡単に説明すると、まず、母点の位置を頂点とし、底面がスクリーンと平行であるような円錐を考える。すると、円錐の表面の奥行きは母点からの距離と見なせる。次に、隠面消去を施しつつ、各母点について、その領域の色で円錐を描画する。図 3 は、隣接した二つの母点 P_1,P_2 を通り、スクリーンに垂直な方向の断面図を示しており、face は円錐の表面である。各画素は depth の判定により塗り分けが可能である。

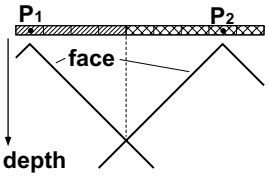


図 3: 概念図 (断面図)

また、[2] において、ハードウェアを用いて、通常の (各点の勢力範囲を示した) ボロノイ図だけでなく、線分や曲線についても考慮した、より一般的なボロノイ図を生成する手法が提案されている。一方、ハードウェアを利用したボロノイ図の応用として、[3] では、ユーザーの指定したエッジに沿うように、小さな正方形タイルを敷き詰めたモザイクの生成が行われている。

[1] 東北学院大学ホームページ (以下の URL) より転載。
http://www.tohoku-gakuin.ac.jp/gakuin/outline/reihai.html

[2] 厳密にボロノイ図の境界線等の座標を計算するのではなく、各画素がどの領域に含まれるか判定して表示したもの。

3 画像のステンドグラス化手法

3.1 提案法の概要

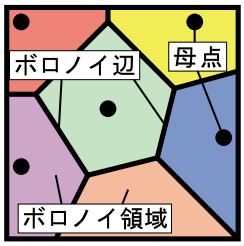


図 4: ボロノイ図

まず、以降の説明で使用する用語を幾つか説明する。ボロノイ図の各部の名称については図 4 を参照のこと。

ボロノイ画像: 生成中のボロノイ図で表現された画像

参照画像: ボロノイ画像生成時に目標とする入力画像

生成されるボロノイ図 (ボロノイ画像) では、各母点は画像中のいずれかの画素の中心に位置しており、また、その母点を含むボロノイ領域は、その母点と同じ位置の参照画像の画素の色で塗りつぶされるものとする。また、ボロノイ画像と参照画像との対応する画素毎の色を比較した場合の誤差の合計 (E_{color} とする) を考え、ボロノイ画像をより参照画像に近づけるための母点操作 (母点の移動) は、E_{color} が減少するように行われる。(一回の) 母点の移動は、隣接する 8 つの画素のいずれかへの移動 (8 方向の移動) を考える。

提案法は大きく二つのプロセスから成る。一つはモデリングプロセスであり、ハードウェアを利用したボロノイ図生成と単純な操作の組み合わせにより、入力画像の特徴 (輪郭等) を表現した、ボロノイ領域の生成を行う。もう一つは、計算して得られた母点やボロノイ辺の情報から、よりステンドグラス風な画像の生成を行なうレンダリングプロセスである。

画像生成プロセスの流れは、以下のようである。

[**モデリング**]

1. ベクトル量子化により、参照画像をインデックスカラー化 (色のテーブルを作成) する
2. ボロノイ図の母点をパラメータ (3.2.2 節参照) に従い配置し、ボロノイ画像を初期化する
3. すべての母点を同時に動かして、参照画像の大局的な特徴をつかむ
4. 各母点を個別に動かすことで、生成中のボロノイ画像の局所的な調節を行う

[**レンダリング**]

1. 母点の座標から通常の (画像精度ではない) ボロノイ図を計算し、ボロノイ辺等の情報を得る
2. よりステンドグラス風な画像を生成するために、**1.** で得られた情報を用いて、幾つかの効果を付加する

以降の節で、各ステップの詳細を説明していく。

3.2 モデリング

3.2.1 ベクトル量子化

各画素の色 (R, G, B) を三次元ベクトルとみなし、ベクトル量子化を行うことで、参照画像中の色のテーブルを作成する。一般に、ベクトル量子化はデータ圧縮を目的とするが、提案法での目的は以下の二点である。

- 参照画像の色数を減らすことで、生成されるボロノイ領域毎の色数を減らす[3]
- 画像同士の各画素毎での色の誤差の計算を高速化 (誤差の値を格納した参照テーブルを作成)

精度よりも計算速度を優先したいので、Lloyd のアルゴリズムを用いた木構造ベクトル量子化 [4] を実装している。色数の指定は、ユーザーに任されている。

3.2.2 母点の初期配置

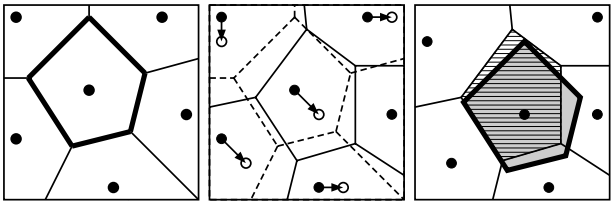
各母点に異なる ID を割り当てて、ハニカム状に母点の初期配置を行う。生成される画像は概ね、この初期配置の精度となる。その際、ユーザーが指定できるパラメータとしては、母点間の間隔と、“ぶれ”(規則的なハニカムの中心点からのずれ) の大きさがある。

3.2.3 大局的な特徴の表現

生成中のボロノイ画像を参照画像に高速に近づけるために、近似的な計算を行ない、すべての母点を同時に動かすということを繰り返す。それによって、参照画像の大局的な特徴を表すようにする。

ここで、「近似的」とあるのは、あるボロノイ領域の形状は、それに含まれる母点と周りの他の母点との位置関係により決定されるもの (図 5(a) 太線) であり、すべての母点が同時に独立に動く場合 (図 5(b)) には、そのボロノイ領域が厳密にはどのような形状になるか (図 5(c) の横線領域) は、その領域に含まれる母点の移動だけでは計算できないためである。そこで、

- ある母点が移動しても (周りの母点の移動は無視して)、そのボロノイ領域の形状は変化しないで、母点と同じだけ平行移動する (図 5(c) の灰色領域)



(a) 移動前の領域　(b) 母点の移動　(c) 移動後の領域

図 5: 近似的なボロノイ領域の計算

と仮定する。このとき、ある母点のある方向への移動に関して、画像中の各画素は、隣接する画素との比較のみで、以下の 4 つの内のいずれかに分類でき、移動した場合の近似的な誤差変化の計算が容易に行なえる。

- 移動してもそのボロノイ領域に含まれたままである
- 新たにそのボロノイ領域に含まれる
- そのボロノイ領域から外れ、周りの領域に含まれる
- 領域外に位置したままで無関係である

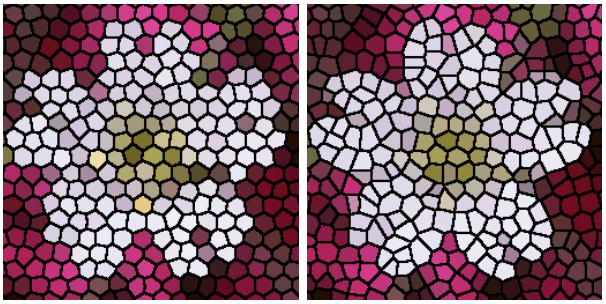
一回のステップは下記のようであり、これを繰り返す。

i. 母点の ID を色と見なして[4]、ボロノイ図を生成

ii. フレームバッファ(ボロノイ画像) の読み出し

iii. 各母点毎に、移動した際の誤差変化を計算

iv. 各母点を最も誤差の減少する方向へと一斉に移動



(a) 初期配置 (3.2.2 節)　(b) 大局的な特徴 (3.2.3 節)

図 6: 各ステップ後のボロノイ画像の変化 (i)

3.2.4 画像の局所的な調節

大局的な特徴を表現する場合とは異なり、ここでは、一度に移動させる母点は一つのみとする。そうすることで、母点が移動した際のボロノイ領域の変化を厳密に計算し、誤差の変化が最小になる方向へと母点を移動させる。ある母点 P を動かす場合の、ハードウェアを使用したアルゴリズムは、以下のようであり、E_{color}(3.1 節参照) の変化がなくなるまで繰り返す。

i. P 以外の母点で構成されるボロノイ図を生成

ii. 各方向への P を移動した後での P のボロノイ領域を、ステンシルバッファに記録 (バッファの値の各ビットを各移動方向に対応させて)

iii. フレームバッファ、ステンシルバッファの読み出し

iv. P の各移動方向毎の誤差変化を厳密に計算

v. 最も誤差の減少する方向へと P を移動

ステンシルバッファを効率的に用いることで、ハードウェアを利用した手法でボトルネックとなり易い、バッファの読み出し回数を減らし、高速化を図っている。

[3] 実際のステンドグラスでも使われるガラスの色数はそれほど多くはない、という観察に基づいている。

[4] ID(24 ビット符号無し整数) の値を、RGB(各 8 ビット) 成分と考えることで、ID を色として扱っている。

3.3 レンダリング

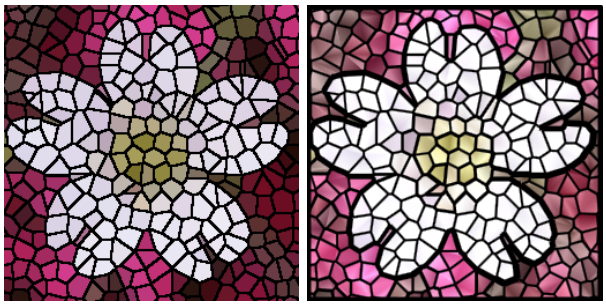
3.3.1 ボロノイ図の計算

モデリングプロセス (3.2 節) で生成されたボロノイ画像からは、ボロノイ辺やボロノイ領域同士の隣接関係を正確に計算できない。そこで、3.2 節で求めた各母点の位置情報から、通常のボロノイ図生成アルゴリズム (逐次添加法を実装している) により、それらを正確に計算する。通常のボロノイ図の生成法については、[5] などで詳しく述べられている。

3.3.2 ステンドグラス風画像の生成

よりステンドグラス風な画像を生成するために、3.3.1 節で得られた情報を用いて、以下の操作を行なう。

- ボロノイ辺 (境界線) の太さを、それに隣接した二つのボロノイ領域の色の差が大きいほど、太くすることで、参照画像中の輪郭等をより強調する。
- 透過光の輝度分布の効果を表すために、各ボロノイ辺の端点で輝度を計算し、ボロノイ領域を塗りつぶす。それにより同一領域内でも色が変化する。



(a) 局所的な調節 (3.2.4 節)　(b) 最終画像 (3.3.2 節)

図 7: 各ステップ後のボロノイ画像の変化 (ii)

4 結果

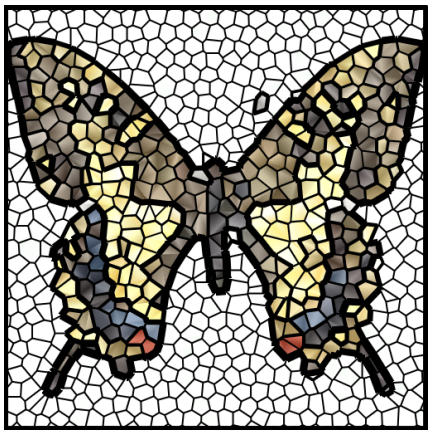
提案法による結果画像を図 8 に示す。それぞれの画像の右側には、画像の名前、画像サイズ (**image size**)、初期配置時の母点間隔 (**grid size**)、母点数 (**points**)、合計の計算時間 (**total time**) を示した。計算時間の測定は、Pentium III 1GHz、GeForce2 Ultra 搭載の PC で行ない、また、グラフィックスライブラリとしては OpenGL を用いた。

結果画像には、図 1 では表せていない、輪郭等の元画像の特徴がよく表現されている。

5 終わりに

ボロノイ図を用いて、元画像の特徴をよく表した、ステンドグラス風画像の生成を行なった。また、グラフィックスハードウェアを利用することで、効率的な画像生成を可能にした。

競走馬
image size
512 × 512
grid size
16
points
1263
total time
60.6 秒



アゲハ蝶
image size
512 × 512
grid size
20
points
825
total time
36.4 秒

アニメ キャラクター
image size
256 × 256
grid size
24
points
160
total time
8.35 秒

図 8: 結果画像とそれらに関する情報

参考文献

[1] OpenGL Architecture Review Board. *OpenGL Programming Guide — The Official Guide to Learning OpenGL, Release 1.* pp.405-406, Addison Wesley, 1993.

[2] K. E. Hoff III, T. Culver, J. Keyser, M. Lin and D. Manocha. *"Fast Computation of Generalized Voronoi Diagrams Using Graphics Hardware."* Proc. SIGGRAPH '99, pp.277-286, 1999.

[3] Alejo Hausner. *"Simulating Decorative Mosaics."* Proc. SIGGRAPH 2001, pp.573-580, 2001.

[4] http://www-rcs.ee.washington.edu/COMPRESSION/

[5] 岡部篤行, 鈴木敦夫. シリーズ〔現代人の数理〕3「最適配置の数理」pp.9-51, 朝倉書店, 1992.