スクウェアパッキングによる石畳テクスチャの生成法

A Method of Generating Pavement Textures Using the Square Packing Technique

宮田　一乗
Kazunori Miyata
東京工芸大学/芸術学部/映像学科
Department of Imaging Art,
Tokyo Institute of Polytechnics

伊藤　貴之
Takayuki Itoh
日本アイビーエム（株）/東京基礎研究所
IBM Research, Tokyo Research Lab.

嶋田　憲司
Kenji Shimada
カーネギーメロン大学/工学部/機械工学科
Mechanical Engineering,
Carnegie Mellon University

1. はじめに

CGで現実感のある画像を生成する技術は、いかにして物体を忠実に表現するかの技術である。物体を特徴付ける要素としては、大きく分けて、物体の形状と、物体表面の色情報や光学属性などの表面属性がある。物体の表面属性を表現する場合、一般的には、バンプマッピングなどのさまざまなマッピング手法がとられている。CGデザイナがマッピングに利用できる素材としては、市販の CD-ROM やインターネット経由で入手できる素材集、およびテクスチャ生成のプログラムから出力されるデータなどがある。

本報告で取り扱う石畳を表現する場合、CGデザイナが路面の敷石を１つ１つモデリングして表現することが理想ではあるが、大変な労力を要するため現実的ではない。一方、石畳の写真素材を路面にマッピングして表現することも可能であるが、以下のような問題が生ずる。

- テクスチャ全体として凹凸が激しいため、光源の位置やカメラアングルの相違によるミスマッチが発生しやすい
- 素材の画像と路面形状のアスペクト比の違いにより、敷石模様に継ぎ目が発生する
- 素材画像の道路形状と表現したい道路形状の不一致により、道路の境界における隙間や敷石の切断が発生する

本報告では、上記のようなさまざまな問題点を解決する１つの手法として、スクウェアパッキングと呼ばれる閉領域の充填パターン生成法を用いて、指定された閉領域を覆う石畳テクスチャを生成する手法を提案する。

本手法では、まず、与えられた道路形状の領域内に、引力と斥力を持つ四角形粒子を発生させて、路面を構成している敷石の充填パターンを生成させる。その後、生成された充填パターンに従って敷石テクスチャを生成し、石畳テクスチャの生成を行っている。

本手法を用いることにより、１つ１つの敷石をモデリングするような手間も要せず、表現したい道路形状の領域を指定して数個の敷石のパラメータを設定することで、デザイナが希望する路面を被覆する、石畳テクスチャの自動生成が可能になった。

2. 石畳と入力データ

本章では、石畳パターンの持つ性質と、石畳テクスチャ生成のために入力するデータ、および過去の類似研究との比較について述べる。

2.1 石畳パターンの性質

実際の石畳の写真[1, 2]を観察して、以下のような傾向を仮定することができた。

- 道路の進行方向に沿って、整列するように石畳が作成されていることが多い。
- おおむね等密度に敷石が詰まっている。

これらの性質を考慮すると、

- 複雑なカーブを描く道路形状に対しても、自動的に等密度で粒子を充填する。
- 指定された道路形状の領域に、うまく沿って整列するように粒子を充填する。

という条件を満たすパーティクルモデルが、石畳の表現に利用できると考えられる。

筆者らは、すでに報告したスクウェアパッキングの手法[3]が以上の条件を満たす手法であることから、敷石パターンの生成過程において適用することにした。一方、与えられた閉領域の分割法としては、例えばボロノイ分割[4]などが考えられるが、この分割法では等方的な分割パターンの生成となり、以上で述べた目的を満たす手法ではない。等方的な石材の敷き詰めをした石畳テクスチャを生成したい場合は、次章以降で述べるパッキングパターンの生成手法の部分を、ボロノイ分割手法に置き換えればよい。したがって、本報告では、敷石の“流れ”が感じられるような、パッキングに異方性を持った石畳テクスチャの生成を目的として述べていく。

2.2 生成のための入力データ

石畳を生成するための入力データとしては、まず、道路の外郭形状と敷き詰める石の平均の大きさ（縦、横、高さ）を幾何データとして与える。また、敷石の大きさのばらつきなどのデ

ータをパラメータとして与える。これらの入力データから、道路の外郭形状に沿った敷石の敷き詰めパターンが生成される。

また、敷石の表面の粗さや色、光学属性などの属性データを別途パラメータとして指定し、さまざまな敷石パターンの生成をコントロールしている。敷石の色と光学属性は、あらかじめテーブルで用意しておき、敷石ごとに乱数で自動的に割り振る形式を取っている。

2.3 過去の類似研究との比較

本手法の類似研究の代表例として、K.W.Fleischer らが提案した、Cellular Texture 生成法[5]（以下 CT 法と略す）が挙げられる。CT 法では、セルプログラムにより性格付けされたパーティクルが、位置や方向性、周囲のパーティクルの影響などを考慮しながら、テクスチャを貼り付ける物体表面に配置される。その後、各パーティクルは棘や半球などの幾何形状に置き換えられ、最終的な物体表面のテクスチャが生成される。

本手法と Cellular Textures 生成法とは、アルゴリズムの大筋では非常に類似している。特に、

・各粒子がエネルギーポテンシャルを持つ。
・各ポテンシャルは、隣接粒子との距離や方向性から算出される。
・ポテンシャルの総計を最小化するための反復処理を行う。

という３つの点では、両手法の概念は一致している。

CT 法が例示しているモデルでは、各粒子のプログラムは隣接粒子への引力や斥力などを明示的に算出しない。ただし、引力や斥力を算出するプログラムを書くこと自体は可能ではある。

一方、本手法では隣接粒子への引力や斥力を陽に算出するので、隣接粒子の移動方向が明確である。この結果として、本手法のほうが収束性の点で有利であると考えられる。実際に、CT 法では計算時間が数時間かかった例があることを論文中で指摘しており、経験的に本手法のほうが計算時間がはるかに短いように思われる。

また、本手法では正方形の粒子を仮定したポテンシャル場を各粒子に与えたが、長方形や五角形、六角形などの正方形以外の形状を仮定したポテンシャル場の記述も容易である。その場合にアルゴリズムの修正は非常に少ない。

一方、CT 法では、粒子の形状が複雑になると、エネルギーポテンシャルの算出式がかなり複雑になることが予想される。この結果として、形状が複雑な粒子であればあるほど、本手法のアルゴリズムの優位性が発揮できると思われる。ただし、この利点は、本報告で取り扱っている石畳ではない、別の物体を表現するときに初めて発揮できるものであると考えられ、この点に関しては、近々報告する予定である。

3. スクウェアパッキングについて

本章では、敷石のパッキングパターンを生成する過程で用いられる、スクウェアパッキング手法について述べる。

3.1 スクウェアパッキングとは

スクウェアパッキング[3]とは、引力や斥力を持つ四角形粒子を、力学シミュレーションによって与えられた領域内部に最密充填する手法である。

本手法には、

(1) 複雑な形状であっても自動的に、隙間や重なりの少ない適度な密度で粒子を充填できる。
(2) ユーザーの意図する任意のベクトル場に沿うように整列させて粒子を充填できる。

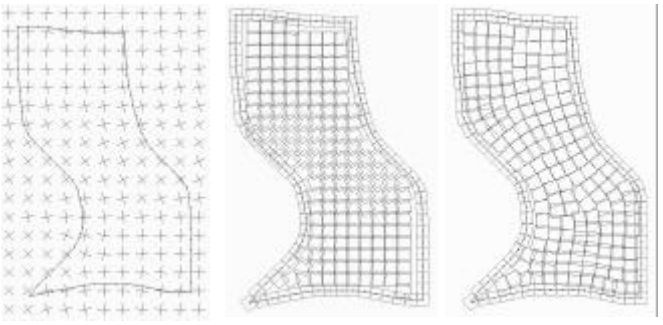
などの特徴がある。したがって、本手法を用いることで、(2) で述べたような石畳の整列状態を表現することができる。なお、本手法はもともと CG のために開発された手法ではなく、有限要素解析のための四角メッシュ生成を目的として開発された手法である。

3.2 スクウェアパッキングの処理の流れ

本手法ではまず、図 1(a)に示すように、領域形状および、粒子の整列方向を表すベクトル場を入力する。

続いて、図 1(b)に示すように、領域境界線上および領域内部に適当な密度で粒子を発生させる。本手法では、領域境界線上の粒子は不動であり、領域内部の粒子は可動であるとする。

続いて、各々の粒子が近接粒子間から受ける引力や斥力の総和を算出し、運動方程式を用いて領域内部の粒子の移動量を算出する。この粒子移動処理を反復することで、粒子が整列方向ベクトル場に沿いながら最密充填した状態を得ることができる。 図 1(c)に、粒子を最密充填した結果を示す。なお、本手法によって得られる粒子の充填結果は、粒子の初期配置によって多少異なるが、ビジュアルな印象には大差ない。

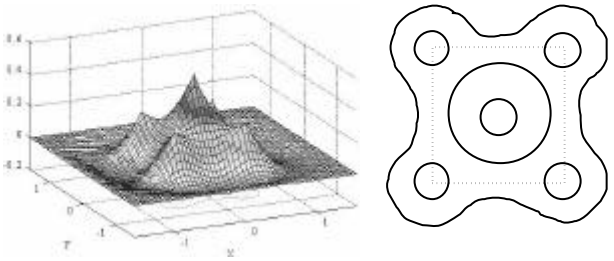


(a) 入力データ　(b) 粒子の初期配置　(c) 最密充填結果

図１ スクウェアパッキングの実行例

本手法では、粒子Ａから粒子Ｂへの斥力を、図2 に示すような、A を原点としたときのＢの位置の関数で算出する。この関数は、四角形粒子の中心点および４頂点に極大点をもつ５個の３次関数を重ね合わせたものである。この関数がもたらす斥力の等値曲線は

四角形に近い形状となり、この四角形の内部まで近隣粒子が接近することを防いでいる。



(a) 3D 表示　　(b) 等値曲線表示

図 2　粒子の引力や斥力の算出に用いる関数

本手法で粒子を整列させるために入力するベクトル場は、ユーザが明示的に与えることもできるし自動生成することもできる。本報告では、形状領域の境界に沿った整列方向を自動算出することでベクトル場を生成している。形状領域の境界線が n 本の線分列 $\mathbf{s}_i\,(i=1..n)$ で構成されているとき、本報告では形状領域内部の任意の点 $\mathbf{p}$ におけるベクトル値 $\mathbf{v}$ を、式 **(1)** で算出する。

$$\mathbf{v}=\frac{1}{n}\sum_{i=1}^{n}\frac{\mathbf{s}_i}{d_i^2} \qquad (1)$$

ここで d_i は $\mathbf{p}$ と $\mathbf{s}_i$ との距離を表す。

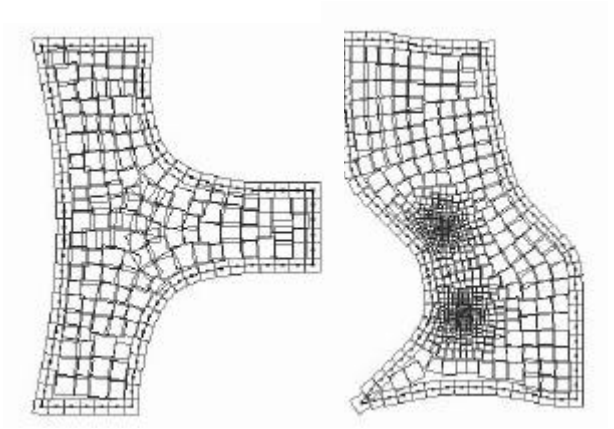


図 3　大きさの異なる四角形粒子を充填した結果。左から(a)粒子の大きさに乱数を乗じたもの、(b)粒子の大きさに位置の関数を用いたもの。

本手法では、粒子の整列方向だけでなく大きさも制御することができる。石畳を構成する石材の大きさは必ずしも一定ではなく、むしろ不規則に変化していることも多い。本報告では、ユーザーの指定する大きさに **0.8～1.2** 程度の乱数を乗じた値を各々の粒子の大きさとしている。図 **3(a)**は、粒子の大きさに乱数を用いて充填した結果である。

さらに、粒子の大きさを位置の関数で表現することも可能である。図 **3(b)** は、粒子の大きさに位置の関数を用いて充填した結果である。ただし、本報告における石畳の画像表現では、位置の関数は用いていない。

スクウェアパッキングの計算時間は、近隣粒子間の力の算出回数に比例する。ここで、粒子の総数を n 、１個の粒子に近隣する粒子の数の平均を m とすると、計算時間は $O(nm)$ となる。ただし m はおおむね定数なので、計算時間は $O(n)$ とみなすことができる。

4.　石畳テクスチャの生成について

石畳テクスチャは、３章で述べたスクウェアパッキングを用いて生成されたパッキングパターンの各閉領域（セル）に対し、以下の３段階の過程により生成される敷石テクスチャを、個別に生成して得られる。

(1) 敷石形状を定義する基本格子の生成と変形
(2) 基本格子のスムージング（丸め）処理
(3) 敷石表面への微細凹凸の付加

以降で、各段階における処理について詳しく述べる。

4.1 テクスチャのデータ構造

本手法では、図 4 に示すように、テクスチャデータをバンプ属性、色属性、光学属性の３つの属性データプレーンに分類して、2次元マトリックスの各セルに保持する。データプレーンの解像度は、レンダリングされて表示されるテクスチャ画像の解像度と等しく設定する。

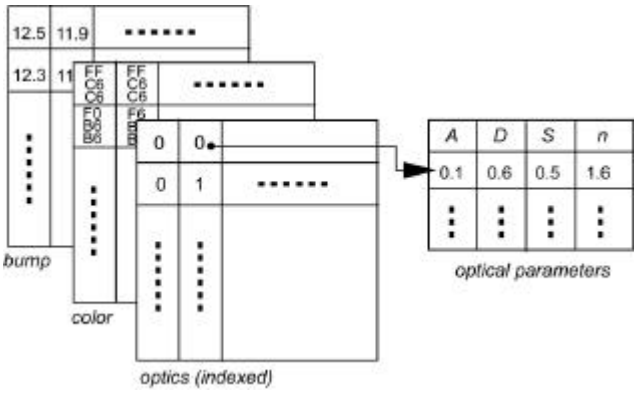


図 4 テクスチャのデータ構造

バンプ属性プレーンは、物体表面の凹凸状態を表し、底面からの変位量、すなわち高さ情報をデータとして持し、4 バイトの浮動小数点データとして保持する。

色属性プレーンは、物体表面の色を表わす。色の表現モデルは多数あるが、本手法では、RGB モデルを用いており、R G, B それぞれ８ビットの色深度を持つ。

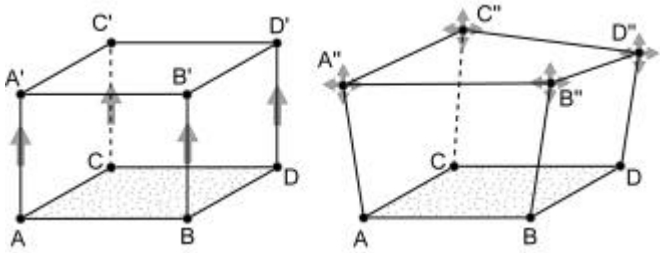
光学属性プレーンは、物体表面の光学属性を表わす。光学属性は、物体を表現する際に用いるシェーディングモデルに依存しており、本手法では Phong のモデルを用いている。光学属性のデータをセルごとに保持するのは非効率的なので、光学属性データテーブルを参照するインデックス形式で、各セルのデータを保持する。

生成された敷石形状と色および光学属性の各データは、各敷石が生成されるごとに、それぞれの属性データプレーンにスキャンコンバージョンされて書き込まれていく。

4.2 基本格子の生成と変形

敷石テクスチャの生成に際して、はじめに、生成されたパッキングパターンに従って、基本格子を生成する。

基本格子は、パッキングパターンの各セルを底面とし、図 5(a)に示すように、与えられた底面を指定された敷石の高さ分だけ垂直方向にスイープした角柱として与えられる。



(a) 基本格子の生成　(b) 基本格子の変形操作

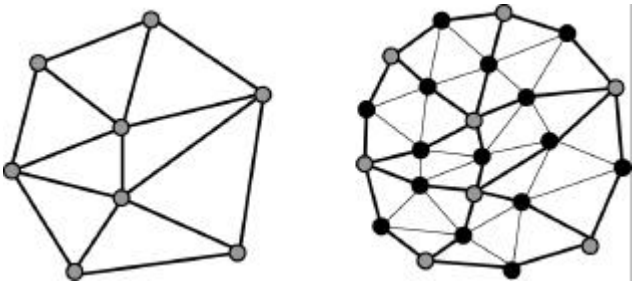
図 5　基本格子の生成と変形

生成された基本格子は、角柱の上面に位置する四隅の頂点 A′B′C′D′を、与えられた変形量の範囲でランダムに移動させて変形される。設定する変形量が大きいと、自然のままに採取されたような敷石が生成され、変形量が少ないと、人工的に形の整えられたような敷石が生成される。このようにして生成された基本格子をもとに、以降の処理で敷石形状が自動生成される。

4.3 基本格子のスムージング

4.2 節で述べた変形後の基本格子を制御格子と想定して、サブディビジョンサーフェイスの生成による角柱のスムージング（丸め）処理を行い、敷石のおおまかな形状を生成する。

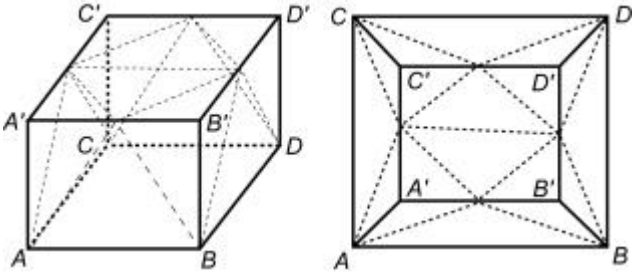
サブディビジョンサーフェイスの生成手法には、Doo-Sabin の手法[6]や、Catmull-Clark の手法[7]、Loop の手法[8]などが報告されている。本手法では、三角形メッシュが生成されることなどの理由から、筆者らの開発環境になじみやすい Loop の手法を用いている。以下に、Loop のアルゴリズムを簡単に述べる。



(a) 与えられた三角メッシュ　(b) 1回分割された三角メッシュ

図 6 Loop の手法による三角メッシュの分割

Loop のサブディビジョンサーフェイス生成のアルゴリズムでは、図 6(a)のように与えられた三角メッシュの各エッジに対し、図 6(b)に示すように新しい頂点（図中の黒丸）が計算され、各エッジは2つに分割される。さらに、新しい頂点同士を結んで、1枚の三角形が4枚の三角形に新たに分割され、分割前の三角形を置き換えていく。新しい頂点の位置は、周囲の頂点の位置とエッジの接続状況などから計算される。



(a) 俯瞰図　(b) 真上から見た接続関係

図 7 基本格子の三角メッシュ

本手法では、変形後の基本格子に対し、図 7 に示すような三角メッシュを生成し、上記の分割を繰り返すことで、基本格子がスムージングされて敷石形状が生成される。

さらに、敷石のタイプに丸石と角石の2つを用意し、丸石を生成する場合は図 7 の三角メッシュを用い、角石を生成する場合は、図 8 に示すように、基本格子のコーナーの部分に頂点を追加して、分割を行う。ここで、図 8 において、灰色の点が追加された頂点であり、コーナーからの各距離は、エッジの長さとの比率として与えられたパラメータから求める。

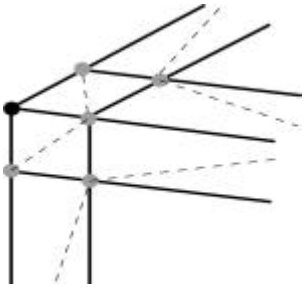


図 8 角石の基本格子(一部)

4.4 微細凹凸の付加

4.3 節で生成された敷石形状の表面に、フラクタルノイズを用いて微細な凹凸形状を付加して、最終的な敷石形状とする。

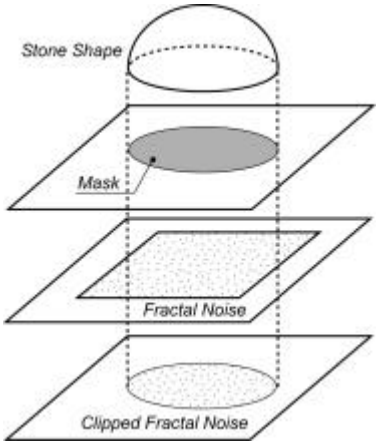


図 9 フラクタルノイズの付加

まず、図 9 に示すように、生成された敷石形状を垂直方向に投影してできる領域、すなわち、敷石形状のデータが存在する領域を、マスクデータとして作成する。続いて、生成されたフラクタルノイズは、このマスクデータでクリッピングされ、敷石形状をバンプ属性のデータプレーンに書き込む時に付加される。

5. 実験結果

以上で述べた手法で生成された、石畳テクスチャのいくつかの例を挙げる。

図 10 は、図 10-1 を基準とした場合、図 10-2 はパッキング粒子の大きさを小さくした例、図 10-3 はパッキング粒子の大きさのばらつきを大きくした例である。

図 11 は、図 11(a)に示すパッキングパターンから生成される、丸石の石畳テクスチャ（図 11(b)）と、角石の石畳テクスチャ（図 11(c)）の例である。

図 12 は、図 12(a)を基準とした場合、図 12(b)は基本格子の変形を 0にした例、図 12(c)は基本格子の変形を大きくした例である。

図 13 に、生成された石畳テクスチャの例を数例挙げる。平均の処理時間は、ペンティアムⅡの300MHzの PC を用いて生成するテクスチャサイズが 512×512 の場合、スクウェアパッキングの処理に約 3 秒、石畳テクスチャを生成するのに約 10 秒であった。

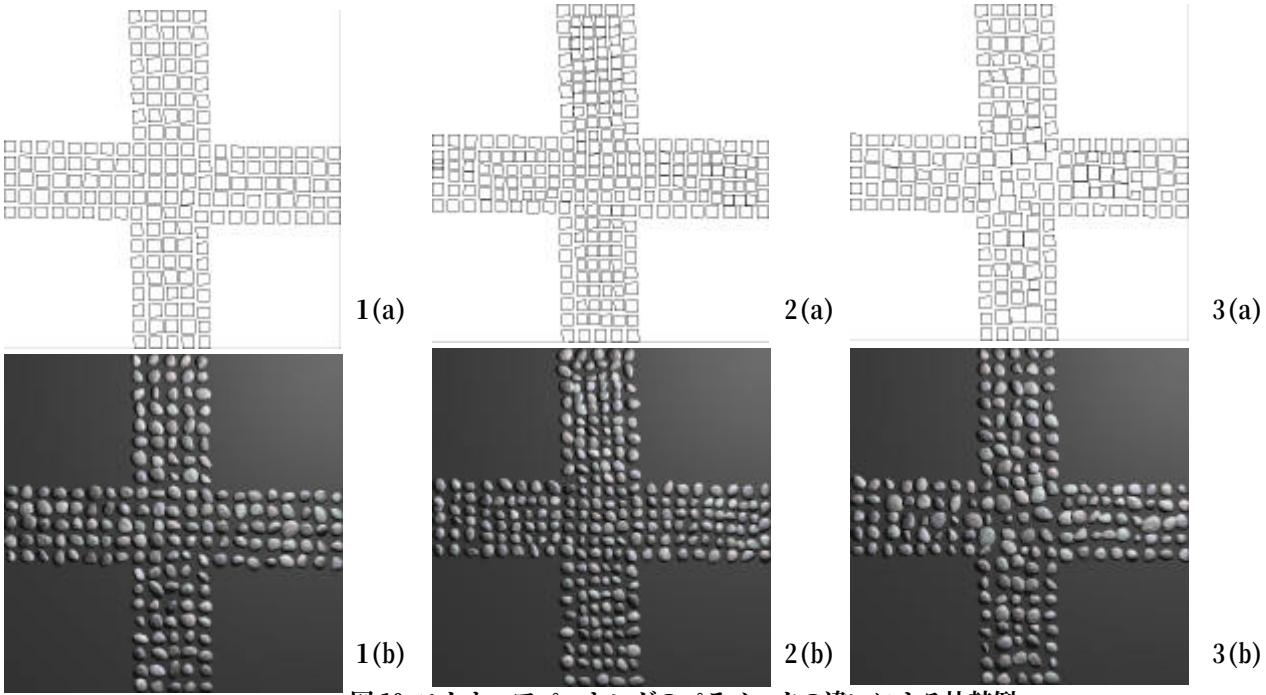


図 10 スクウェアパッキングのパラメータの違いによる比較例

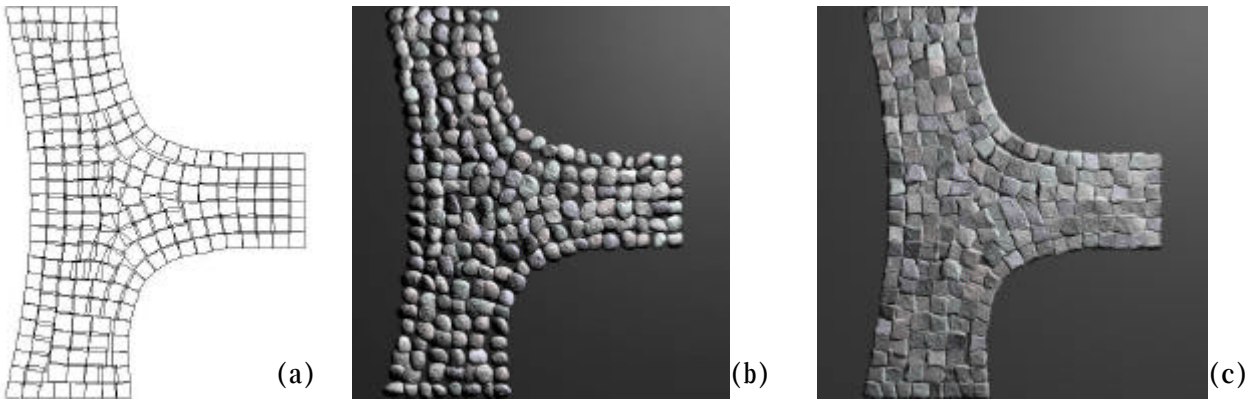


図 11 敷石のタイプの相違

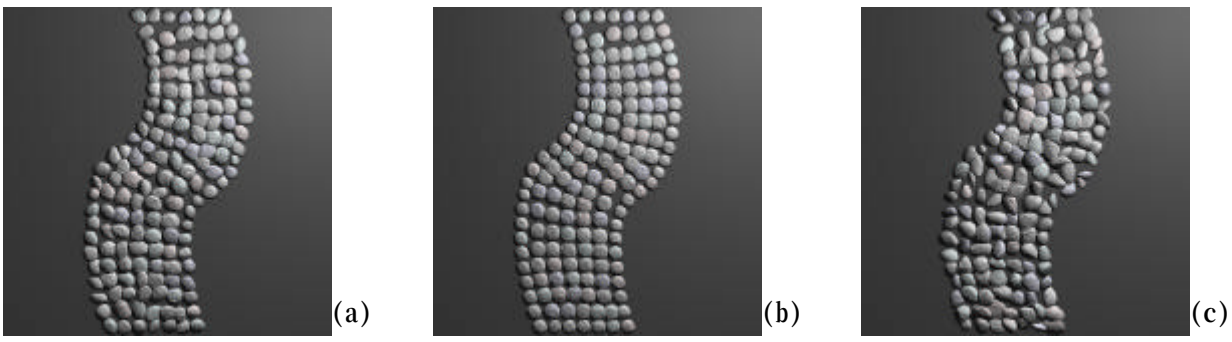


図 12 基本格子の変形率の違い

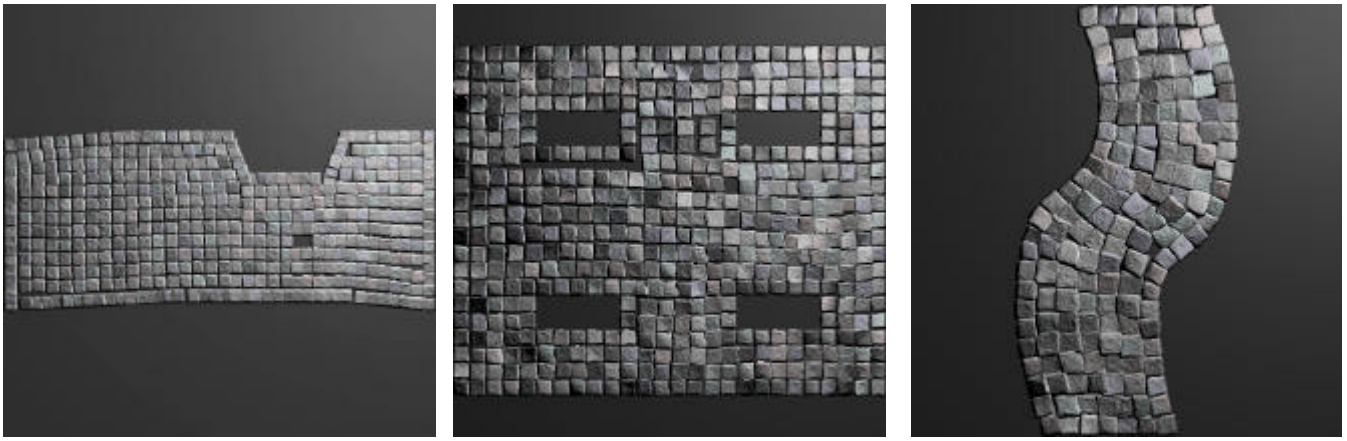


図 13 さまざまな石畳テクスチャの例

6．おわりに

外郭形状と数個のパラメータを入力して、自動的に石畳テクスチャを生成する手法を報告した。

ユーザーが、よりデザインしやすいシステムにするには、初期入力した道路形状を編集することで、石畳テクスチャを編集できるようなシステムへの変更が必要である。これは、ユーザーインターフェイスを組み替えることで可能であると考えている。

今後は、本手法を有機的なテクスチャの生成法に応用していきたい。

参考文献

[1] 重森三玲, "庭、神々へのアプローチ, "
誠文堂新光社, 1976.

[2] http://www2.wbs.ne.jp/~ oamack/photos/toukaido.htm

[3] 伊藤他, "自動四角メッシュ生成手法の検討, "
シミュレーション, Vol.18, No.2, pp.19-25, 1999

[4] 杉原厚吉, "パターン認識の道具としてのボロノイ図構成算法の整備, "情処研報-グラフィックスと CAD, Vol.89, No.16, pp.1-8, 1989.

[5] K.W.Fleischer, et.al, "Cellular texture generation, " Proceedings of SIGGRAPH '95, pp.-. 1995.

[6] D. Doo and M.Sabin, "Analysis of the behavior of recursive division surfaces near extraordinary points, "Computer Aided Design, Vol.10, No.6, pp.356-360, 1978.

[7] E.Catmull and J.Clark, "Recursively generated B-spline surfaces on arbitrary topological meshes, " Computer Aided Design, Vol.10, No.6, pp.350?355, 1978.

[8] C.Loop, "Smooth subdivision surfaces based on triangles, "Master 's thesis, University of Utah, Department of Mathematics, 1987.