任意形状をもつオブジェクトの配置によるテクスチャ生成
- A Method of Generating Texture via Distributing Arbitrary Elements -

櫻井 快勢　宮田一乗

Kaisei Sakurai and Kazunori Miyata

北陸先端科学技術大学院大学

Japan Advanced Institute of Science and Technology

E-mail: {sakurai, miyata}@jaist.ac.jp

1. はじめに

テクスチャは，コンピュータグラフィックスにおいて，高品質なレンダリング結果を得るために用いられ，多くの制作者がその表現に力を注いでいる．表面の模様は，材質の設定とは独立して，テクスチャで設定するため，使用するテクスチャがレンダリングの品質を決める．

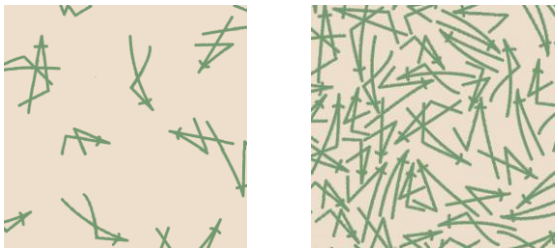
テクスチャの生成では，模様の要素となるオブジェクトの配置が品質を決めるひとつの要素となる．このようなテクスチャの制作で，非周期的なオブジェクトの配置を要する場合は，単純なタイリングが使えないため，作業が多くなり，負担が大きい．私たちは，この非周期的な模様の制作に関して，作業量を減らす仕組みを提供する．本研究では，重ならない領域を指定したオブジェクトと出力の画像サイズ，計算に必要なパラメータを入力し，それらのオブジェクトを自動的に，かつ，非周期的に配置することを目的とする．ここでは，単純なランダムではなく，重ならない領域を指定することで，使用者が配置を操作できるようにする．非周期的なオブジェクトの配置法として，配置済みのオブジェクトの位置関係から次に配置する位置を決定する方法が考えられる．点の配置法ではあるが，Dunbar と Humphreys は，配置済みの点から，近隣点が存在する領域を推定し，高速な Poisson-disk distribution を提案した[1]．これは Poisson-disk distribution では，点間の距離がある程度決まっているために実現した手法である，任意の形状を持つオブジェクトでは，形状に応じた中心点間の距離や配置の能否を計算する必要があるため，点の分布のように適切な配置位置を推定できない．一方，重ならない領域を指定する方法では，接触判定するのみであり，試行数は増えるが，点の分布と同様に配置位置を定めることができる．ここでは，問題を簡単にするために，2 次元を対象にする．本手法は，日本の伝統文様や敷き詰めなどに適用することができる．

非周期的にオブジェクトを配置するには，物理シミュレーションの衝突とそのリアクションを応用しても，実現可能に思える．しかし，以下の理由から，手続き的なテクスチャ生成の方が適切と判断した．多量のオブジェクトに対して，衝突判定とそのリアクションを考慮したシミュレーションを行うと，多くの衝突が起こり，オブジェクトの位置が著しく変化し続ける．そのため，定常状態にならず収束しない可能性がある．また，移動の係数に粘性などを含め，移動量を減らすことで状態の著しい変化を抑えることはできるが，同様に定常状態になるとは限らない．対照的に，手続き的なテクスチャ生成では，終了条件を明確にするため，確実に所望のテクスチャを得ることができる．

2. 関連研究

これまで，テクスチャ生成に関する手法が数多く提案されている．テクスチャを自動的に生成する手法は，一般的に「プロシージャルテクスチャ」と呼ばれる．特に， Perlin が提案したテクスチャ生成手法は幅広く活用されており[2,3]，パン[4]や煙，布地，岩肌などの表現が可能である[5]．ほかにも，Worley が提案した細胞のような形状を生成する Cellular texture も活用されている[6]．いずれの手法もランダムな関数に依存するため，任意の形状を生成することができない．一方，石垣[7]や布地[8]，生物の模様[9,10]などの特定のテクスチャに特化したテクスチャ生成が提案されている．ただし，任意のオブジェクトを指定することはできないため，生成対象の特徴を反映させるためには，生成アルゴリズムを開発する必要があり，開発コストが高い．

本手法と関連の深い，オブジェクトを分布してテクスチャを生成する手法として，Lagae と Dutre は Poisson-disk distribution を応用したテクスチャ生成法を提案した[11]．この手法は，Poisson-disk distribution で配置した各点上に一つのオブジェクトを配置する手法である．これらの点間は一様であるため，図 1(a)に示すようにオブジェクト間の距離がほぼ一定となる．そのため，長い形状や凹部がある形状のオブジェクトでは密に配置されない箇所が出てくる．一方，本手法では，距離に依存しないため，図 1(b)に示すように既存手法[11]以上に，密なテクスチャを生成することができる．



(a) Lagae と Dutre の手法　　(b) 本手法

図 1 松葉オブジェクトでの手法による違い

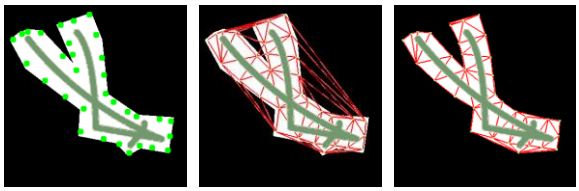
また，与えられたサンプルを解析し，その情報を用いて新しいテクスチャを生成するテクスチャシンセシスと呼ばれる技術が提案されている．テクスチャシンセシスには，主にピクセルベースとパッチベースのふたつのアプローチがある[12]．ピクセルベースのテクスチャ生成[13,14]は，サンプル内の各ピクセルとその周辺ピクセルの色情報から新しいテクスチャのピクセルの色を決定する．パッチベース[15,16,17,18,19]は，エッジ抽出などでサンプル内のオブジェクトごとに分け，オブジェクトごとの相対位置を保持したまま新しいテクスチャにオブジェクトを配置する．いずれの手法も，密度などのオブジェクトの位置関係を保持するような仕組みであり，密度を変更することはできない．模様の生成において，密度の制御は必須であることから，テクスチャシンセシスの手法では不十分であると考える．

その他のテクスチャ生成では，Kaplan と Salesin が提案したオブジェクトを敷き詰め繰り返しパターンを生成する Escherization[20,21] や Kim と Pellacini が提案した入力の画像の色に合うオブジェクトで敷き詰める Jigsaw Image mosaics[22]などが挙げられる．Escherization は，非周期的なテクスチャは生成できず，Jigsaw Image mosaics は，入力画像の色と入力オブジェクトの差異をコスト関数として最小化するため，色数が多いオブジェクトを配置ができず，本研究の目的を達成できない．

3. 配置用オブジェクトの生成

本手法では，非周期的な分布を行う際に，配置済みのオブジェクトと接触しない排他的な領域を指定する．排他的な領域は，一つのオブジェクトにつき一つ指定する．オブジェクトは，2 次元の画像を用いて指定する．画像には，配置したい模様と排他的な領域を指定するための頂点を指定しておく．

排他的な領域は，2 次元の三角形メッシュで構成する．メッシュを用いることで，入力データにラスタ形式とベクタ形式のどちらも扱えるようになり，かつ，任意の形状を構成することができる．さらに，今後，3 次元の拡張が容易になるほか，変形の処理も対応できるようになる．使用者は，三角形メッシュの頂点とオブジェクトの外部の色を指定することで，任意の形状を指定する．図 2(a)では，緑の点が頂点を表し，外部の色は黒を指定した．図 2(a)中の白い領域は，背景色であり，これも排他的な領域の内部として定義する．入力した点を母点としてドロネー図を構成し，図 2(b)のように，三角形メッシュを構成する．このままでは，凸包のメッシュであるため，図 2(c)に示すように，重心が外部に位置する三角形をメッシュから削除する．



(a)　　(b)　　(c)

図 2 オブジェクトの生成 (a) 指定されたオブジェクトと頂点 (b) ドロネー図 (c) オブジェクトと排他的な領域を構成する三角形メッシュ

ドロネー図は，与えられた点群にて，比較的近い点で三角形を構成する．そのため，与える母点間の距離にムラがある場合，所望のオブジェクトを構成できないことがある．このとき，ムラができないように，再度，使用者が頂点を追加入力して，排他的な領域を再構成する．

4. オブジェクトの配置

テクスチャは一般的に矩形であるため，本手法では矩形領域にオブジェクトを配置する．使用者は，複数のオブジェクトに排他的な領域を付加して入力する．

配置済みのオブジェクトの排他的な領域と接触しないように，オブジェクトを逐次矩形領域に配置する．基本的には Poisson-disk distribution のダートスローイング法[22,23]と同様で，ランダムに選択した位置にオブジェクトを配置し，配置済みのオブジェクトと接触したら，配置を取りやめるという手続きを終了条件まで繰り返す．単純に配置したのでは，オブジェクトの方向がそろってしまうために，オブジェクトを回転させる処理を加え，視覚的な一様さを与える．

上述した処理は，次の 3 ステップとなる．また，図 3 に手続きの概略を示す．

(1)配置するオブジェクトを選択

(2)オブジェクトを指定した領域内に配置

(3)オブジェクトを回転

ただし，ステップ(2)と(3)では，排他的な領域の接触により，その配置を取りやめることがある．それぞれの処理を 4.1-4.3 節で後述する．以上の処理を 4.4 節に記述する終了条件を満たすまで繰り返す．

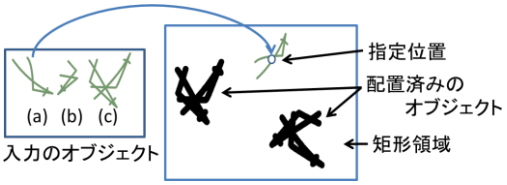


図 3 概略図. 入力のオブジェクトからオブジェクトを選択し，指定位置に配置する. その後回転させ，配置済みのオブジェクトと接触しないならば配置終了

4.1 配置するオブジェクトの選択

複数のオブジェクトを配置するために，配置の順番を決める必要がある. 本手法では以下のことを考慮して，入力順にオブジェクトを選択する.

本手法では，オブジェクト同士が接触しないように配置する. そのため，配置されているオブジェクトの隙間に，あるオブジェクト A を配置するとき，A の大きさに依存して配置の能否が決まる. ここで，オブジェクト A が大きく，オブジェクト B が小さいとし，これらをランダムに選択することを考える. ランダム選択では，A は配置できず，B が配置できる隙間のみになったとしても，A が続けて選択され，配置に無駄が生じる場合がある. また，B が続けて選択されると，A が入る隙間がなくなることが考えられ，複数のオブジェクトを入力したにも関わらず，B のみが配置される. これらの配置の偏りを避けるために，オブジェクトを入力した順に選択することで，一様な選択を実現する.

各配置にて，取りやめの処理があるが，取りやめた場合も次の配置には，次のオブジェクトを選択する. たとえば，3 つのオブジェクト abc が入力されたとすると，abcabcab…と選択を繰り返す. そして，オブジェクト a にて配置の取りやめがあった場合，次の配置では，オブジェクト b を選択する.

4.2 オブジェクトの配置位置

本手法の基本的な手続きは，Poisson-disk distribution のダートスローイング法[22,23]と同様で，オブジェクトの接触を判定し，配置の能否を決定する. ダートスローイング法は，一様な点を配置するための手法であり，ひとつの点を配置するとともに，円状の排他的な領域を配置する. 以降，円同士が接触しないように点を配置する. もし円状の領域同士が接触した場合は，その配置を取りやめる. 本手法では円状の排他的な領域の代わりに，3 章で生成した配置用のオブジェクトの排他的な領域を用いる.

本手法では，オブジェクトを配置可能な領域を計算し，その中に配置する. その計算のために，あらかじめ，矩形領域内に Poisson-disk distribution にて一様にサンプリング点を配置する. 図 4 に示すように，オブジェクトが配置されるごとに，そのオブジェクト内にあるサンプリング点を削除する. そのため，残っているサンプリング点上は配置可能な領域を示す. 具体的な配置処理は，オブジェクトのバウンディングボックスの中心がサンプリング点の位置になるように移動させる. 配置ごとに領域が減っていくため，計算量は $O(n \log n)$程度になる. このときサンプリングの粒度は，小さいほうが望ましい. 筆者らの実験では，Poisson-disk distribution で指定する半径を配置するオブジェクトのバウンディングボックスの対角線の長さを考慮して決定する. 具体的には，複数の入力されるオブジェクトの中で，最も短い対角線の長さの 10%を半径とした.

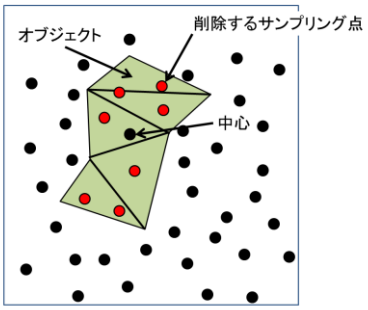


図 4 サンプリング点とオブジェクト

4.3 節で後述する処理で，オブジェクトの接触を解消するために，オブジェクトを回転させるが，一回転してもなお，オブジェクトが接触する場合には，そのオブジェクトの配置を取りやめる. ただし，一回転しても接触し続けることを事前に検知できる場合があり，その場合には配置を取りやめ，次の配置に移る.

接触が続くことを検知するために，まず，配置済みのオブジェクトとの接触の位置について考える. このとき，オブジェクトの境界上で，最も中心に近い点（以後，境界上の最近傍と呼ぶ）に注目する. 図 5 (a)のように，配置済みのオブジェクトが境界上の最近傍に重なっていない場合，回転によって接触が解消される可能性がある. 一方，図 5 (b)のように，配置済みのオブジェクトが境界上の最近傍に重なっているとき，接触が解消される可能性はない. 解消されない理由は以下である. 境界上の最近傍は，その定義から最も近い点であり，境界上の最近傍の回転の軌跡は，いかなる回転をしても，常にオブジェクトに内包されている. 例として，図 5(b)に示した境界上の最近傍の軌跡がオブジェクトに内包されていることがわかる. この軌跡の内部の領域を，円領域と呼ぶ. この円領域は，回転しても常に同じ位置にあり，常にオブジェクトに内包されているため，円領域に接触している場合，回転によって接触を解消することはできない.

円領域の中心は回転の中心であり，円の半径は，中心からオブジェクトの境界上の最近傍の点との距離である. 境界上の最近傍は，オブジェクトを構成する三角形において，内外をわける辺上にある. 具体的には，図 6 が示すように，辺が2三角形で共有している場合は，その辺はオブジェクトの内側とし，

ひとつの三角形である辺は内外をわけることを示す．すべての内外をわける辺と中心との距離を求める．この距離が円の半径とする．この領域は，回転の中心がオブジェクトの内側にあるときにのみ有効である．

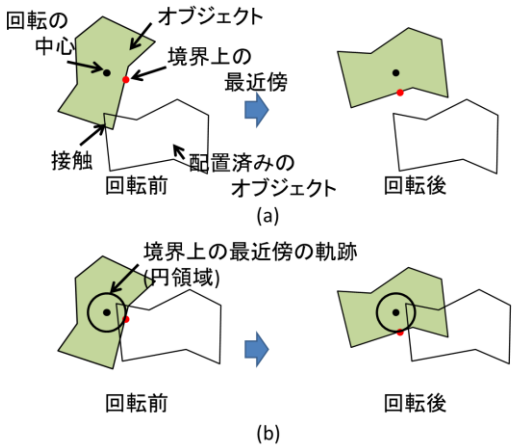


図 5 回転による接触解消の能否の模式図．(a) 解消できる例 (b) 解消できない例

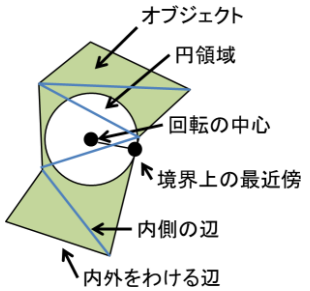


図 6 配置を取りやめるための円状領域

4.3 オブジェクトの回転

ダートスローイング法で扱う「点」には方向がないが，本手法で扱うオブジェクトは 2 次元であるため，方向がある．方向がそろうと視覚的な異方性が出るため，回転を加える．回転の中心となる点は，オブジェクトのバウンディングボックスの中心点とする．このときの回転角 θ は $0°<\theta<d(0°<d<360°)$の範囲で定義され，この範囲内でランダムに決定する．また，d を変化させることで視覚的な一様さを制御する．回転にはあらかじめ分解能 $\Delta\theta$ を設定しておき，それに従い回転させる．筆者らの実験では，分解能 $\Delta\theta$ を 5° とした．分解能と回転角 θ の関係は，変数 r で表現すると $\theta=r\Delta\theta$ となる．ただし，r は $0 \leqq r<d/\Delta\theta$ の自然数とする．

具体的な処理として，まず，4.2 節で先述した配置の直後に，オブジェクトに対して $0° \leqq \theta<d$ の範囲内でランダムに初期の回転角を与える．この回転角は，変数 r にランダムな値 r_0 を代入した $r_0\Delta\theta$ とする．このとき，配置済みのオブジェクトと接触していなければ，このオブジェクトの配置を終了とし，次の配置に移る．一方，この配置にて，配置済みのオブジェクトと接触している場合には，変数 r に 1 を加算しながら，接触しない回転角を探す．ただし，図 7 に示すように，どこか一部でも接触が続くような場合には，この位置での配置を取りやめる．r の範囲は，$0 \leqq r<d/\Delta\theta$ であるため，$d/\Delta\theta$ に到達した時点で，r に 0 を代入する．r が r_0 になったとき，この位置で接触を避けられないことがわかる．

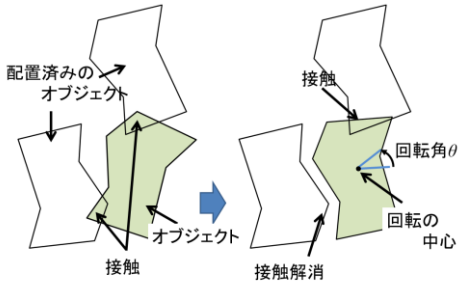


図 7 回転で接触が解消しない例

4.4 終了条件

本手法では，連続して配置が取りやめられた回数が指定数を超えたときに，オブジェクトの新規配置を終了する．配置の取りやめは，4.2 節と 4.3 節で述べた処理である配置の位置の決定と接触を解消するための回転で起こる．オブジェクトが配置された場合は，この回数をリセットする．取りやめが起こる原因が，配置の位置が不適切であり，別の位置ならば配置が可能な場合がある．そのため，一度の取りやめで手続きを終了すると所望の結果を得られない．そのため，取りやめの回数を指定させ，その回数を超えた時点で終了とする．この取りやめの回数は使用者が指定する．

また，複数のオブジェクトを入力している場合は，順番に配置の機会が回ってくるため，すべてのオブジェクトに一通り，配置をさせることが妥当である．たとえば，3 つのオブジェクト abc を入力したとき，取りやめの回数を 2 回と指定した場合，ab の取りやめが起こった時点で終了する．c が配置可能であるに関わらず，配置されないため，所望の結果が出力されない．それを回避するために，回数の指定は，入力の数を考慮する．

これらを踏まえて，筆者らの実験では，入力の数 x100 回を指定している．

5. 結果

本手法では，配置するオブジェクトと出力のサイズ，および，制御パラメータとして，配置に用いるサンプリング点の Poisson-disk distribution の半径，回転角度の分解能，回転角度の範囲を与える．入力のオブジェクトには，排他的な領域を生成するための頂点と外部の色を指定する．図 8 に入力

のオブジェクトと生成結果を示す. 図 8 (a)(c)(e)では, 回転角の範囲dを 360°, 図 8 (b)(d)(f)では, 範囲dを 0°を設定した. また, 図 8 では, 入力のオブジェクトに合わせた背景色を設定した. これらの結果から, 任意の形状を持つオブジェクトの配置が可能で, かつ, 方向を制御できることがわかる. また, 図 9 に, 排他的な領域を変更した例を示す. この結果から, 密度の制御ができることがわかる.

表 1 に計算時間と配置したオブジェクトの数を示す. この実験は, 配置のアルゴリズムをVisual C++で実装し, Intel Core i7 (3.07 GHz) と 12GB RAMを搭載したWindows PC上で実行した.

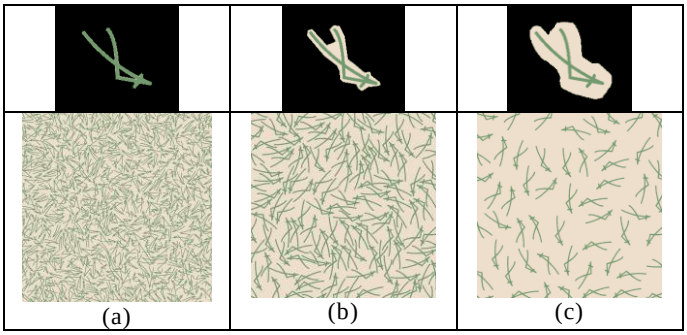


図 9 排他的な領域の変更による密度の操作.
上行 入力のオブジェクト. 下行 出力結果.

表 1 配置数と実行時間

名前	配置数	時間[s]
図 8 (a)	186	15.366
図 8 (b)	156	4.368
図 8 (c)	106	3.541
図 8 (d)	90	1.279
図 8 (e)	24	0.593
図 8 (f)	26	0.655
図 9 (a)	127	21.575
図 9 (b)	98	6.411
図 9 (c)	57	2.044

6. まとめ

本手法により, 非周期的なオブジェクトの配置が可能となった. また, 重なりを避ける領域を指定することで, 配置の制御が可能とした. 汎用性も高く, テクスチャ生成として, 有効な手段と考える.

今後は, 3 次元の拡張, もしくは, さらに汎用性を高めるために, 変形を考慮した配置に展開したい.

参考文献

[1] Dunbar D., Humphreys, G. “A spatial data structure for fast Poisson-disk sample generation”, ACM Trans. Graph., 25, pp. 503 - 508, July 2006

[2] Perlin K., “An image synthesizer” SIGGRAPH Comput. Graph, 19, pp.287 - 296, July 1985.

[3] Perlin K., Hoffert E. M., “Hypertexture”, SIGGRAPH Comput. Graph., 23, pp. 253 - 262 July 1989

[4] Xenakis A., Tomson E., “Shading food: making it tasty for ratatouille”, ACM SIGGRAPH 2007 courses, 2007

[5] Ebert D. S., Musgrave F. K., Peachey D., Perlin K., Worley S., Texturing and Modeling, Third Edition: A Procedural Approach, Morgan Kaufmann, 2002

[6] Worley, S., “A cellular texture basis function”, Proceedings of the 23rd annual conference on Computer graphics and interactive techniques, pp. 291 - 294, 1996

[7] Miyata, K., “A method of generating stone wall patterns”, SIGGRAPH Comp. Graph., 1990

[8] Adabala N., Magnenat-Thalmann N., “A Procedural Thread Texture Model”, journal of graphics, gpu, and game tools, 8, pp. 33 - 40, 2003

[9] Witkin A., Kass M., “Reaction-diffusion textures”, SIGGRAPH Comput. Graph., 25, pp. 299 - 308, 1991

[10] Turk G. “biological models, reaction-diffusion, texture mapping”, SIGGRAPH Comput. Graph., 25, pp. 289 - 298, 1991

[11] Lagae, A. , Dutre P., “A procedural object distribution function”, ACM Trans. Graph., 24, pp. 1442 – 1461, 2005

[12] Wei L.-Y., Lefebvre S., Kwatra V., Turk G., “State of the Art in Example-based Texture Synthesis”, Eurographics 2009, State of the Art Report, EG-STAR, 2009

[13] Efros A., Leung T., “Texture synthesis by nonparametric sampling”, In International conference on Computer Vision, pp. 1033- 1038, 1999

[14] Wei L.-Y., Levoy M., “Fast texture synthesis using tree-structured vector quantization”, SIGGRAPH Comput. Graph, pp. 479 – 488, 2000

[15] Hurtut T., Landes P.-E., Thollot J., Gousseau Y., Drouillhet R., Coeurjolly J.-F.: “Appearance-guided synthesis of element arrangements by example” NPAR ’09, ACM, pp. 51–60, New York, NY, USA, 2009

[16] Liu Y., Wang J., Xue S., Tong X., Kang S. B., Guo B., “Texture splicing”, Computer Graphics Forum, 28, 7, pp. 1907–1915, 2009

[17] Dischler J.-M., Maritaud K., Levy B., Ghazanfarpour D., “Texture particles”, COMPUT. GRAPH. FORUM, 21, pp. 401–410. 2002

[18] Ijiri T., Mech R., Igarashi T., Miller G.: “An example-based procedural system for element arrangement”, COMPUT. GRAPH. FORUM, 2008

[19] Barla P., Breslav S., Thollot J., Sillion F., Markosian L. “Stroke pattern analysis and synthesis”, In Computer Graphics Forum, 25., 2006

[20] Kaplan C. S., Salesin D. H. “Escherization”. SIGGRAPH Comput. Graph, pp. 499–510. 2002

[21] Kaplan C. S., Salesin D. H. “Dihedral escherization”.

In Proc. of Graphics Interface 2004, GI '04, pp. 255–262, 2004

[22] Mitchell D. P., "Spectrally optimal sampling for distribution ray tracing" SIGGRAPH Comp. graph., 1991

[23] McCool M., Fiume E., "Hierarchical poisson disk sampling distributions", In the conference on Graphics interface '92, 1992

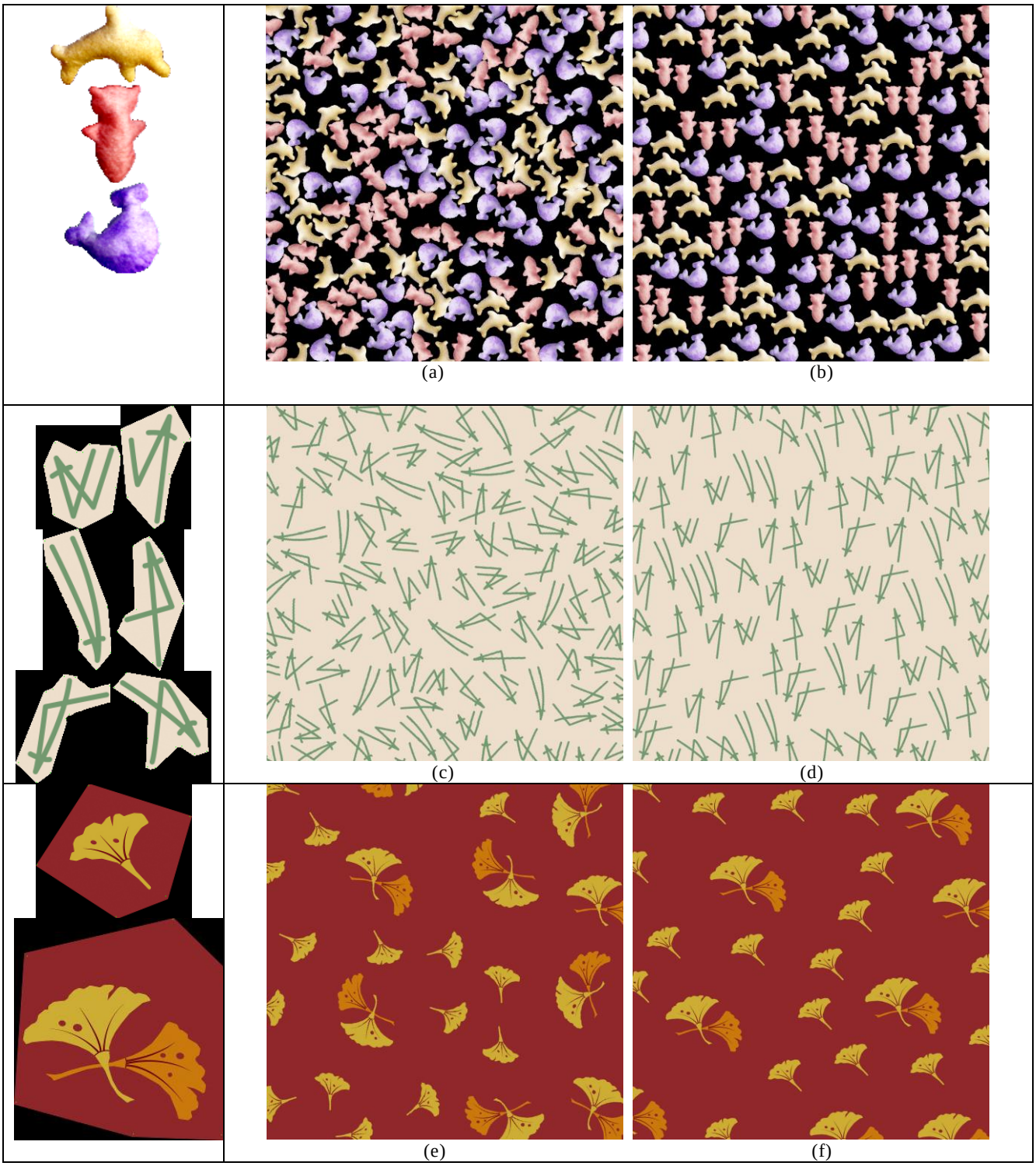


図 8 出力例 左列 入力のオブジェクト，右列 出力のテクスチャ．
(a) (c) (e) は視覚的に一様になるように回転処理．(b) (d) (f) は回転せずに配置．