K 近傍ギブスサンプラによる確率的テクスチャ合成

上瀧　剛†　内村　圭一†

† 熊本大学大学院自然科学研究科, 〒 860–8555 熊本市黒髪 2–39–1
E-mail: †{koutaki,uchimura}@cs.kumamoto-u.ac.jp

あらまし　確率的ノンパラメトリックなテクスチャ画像合成モデルおよびテクスチャ画像合成アルゴリズムを提案する．提案手法はギブスサンプラを用いて出力画像を反復的に更新することで，従来の貪欲法的なテクスチャ画像合成手法に比べて局所解に陥りにくく自然な結果が得られる．また，提案モデル式より，処理時間のボトルネックが学習画像の画素サイズに依存する点を指摘し，この部分を Kd 木を用いた K 近傍探索によって効果的に近似する方法を提案する．実験では，従来の手法と比べて目視で分かるほどの不自然な領域が発生せず，学習画像に忠実なテクスチャ画像を合成できることを示す．また，提案した近似モデルが近似しない場合と比べて，ほぼ同等の品質が得られ，処理時間に関しては 5 倍程度高速化できることを示す．

キーワード　テクスチャ合成, マルコフ確率場, ギブスサンプラ,Kd 木

1. ま え が き

コンピュータグラフィクス（ＣＧ）において，３Ｄモデルにテクスチャ画像を張り付けるテクスチャマッピングはリアルな物体表現に必要不可欠である．そこでは，３Ｄモデルにシームレスに張り付けられるように，画像の上下左右端でトーラス状に画像の濃淡値が連続するテクスチャ画像を用意する必要がある．すなわち，単純に実物のテクスチャを撮影した画像（実テクスチャ画像）は前述の条件をみたさずそのまま用いることができない．そこで，実テクスチャ画像からトーラス状に画像の濃淡値が連続するテクスチャ画像を合成する研究が進められている [1] [3] [4].

テクスチャ画像に関する研究は，古くからパターン認識用途で行われてきた [3] [5] [6] [7]．共起行列を用いた統計的なテクスチャ解析法や [5]，各画素が周辺の画素に確率的に依存するというマルコフ確率場モデル [7] などがある．中でも，マルコフ確率場モデルを用いたテクスチャ解析法はモデルが単純で汎用性が高く，数多くの研究がなされている．例えば，２値画像に対するマルコフ確率場モデルは統計物理におけるスピングラス問題との類似しており [8]，マルコフ連鎖モンテカルロ法によるサンプリング操作によってテクスチャ画像を合成可能である．同様に多値画像に対しても，ガウシアン・マルコフ確率場モデル [9] や平均場近似モデル [10] などが提案されている．これらのマルコフ確率場モデルは，注目画素と周辺画素が少数のパラメタからなる正規分布 [9] や２項分布 [7] に従うとしたパラメトリックモデルで定式化される．

パラメトリックモデルに対して，最大擬似尤度推定などを用いて学習画像から推定したパラメタを用いて，布地や砂などのテクスチャ合成に成功した報告がある [9]．しかし，このモデルは木目やタイルなどといったパターンのテクスチャ合成は困難である．なぜならば，これらのパターンは構造的な長周期のパターンや多種類・多方向の局所パターンが混在しており，少数のパラメタで駆動する前記パラメトリックモデルではこのような多峰的な画素分布を持つパターンを十分に表現しきれないためである．

これに対して，近年，予め学習画像を張り合わせることでテクスチャ画像を合成するノンパラメトリックなテクスチャ合成法が注目を浴びている [4] [1] [11]．これらの手法は学習データを大量に扱うためメモリおよび処理時間に多くを要するものの，パラメトリックモデルでは困難であった自然物のテクスチャ合成も可能である．また，ネックになっていたメモリや処理時間に関しても計算機の発達により大きな障害にならなくなっている．

Li-Yi らは学習画像から切り出した画素を張り付けてテクスチャ画像を合成する方法を提案している (以降, WL 法と呼ぶ) [1]．この手法は学習画像と出力画像との最小二乗誤差となる画像パッチ（例えば 3×3 画素集合）の中心画素を順次張り合わせる単純な方法であるが，自然物のテクスチャを高速に合成可能である．Efros らは大きなブロック画像を張り合わせた後に，張り合わせたブロック画像同士の重複する領域内での繋ぎ目が目立たないような画素を選択するパッチベースのテクスチャ合成方法を提案している [11]．これらの手法は決定論的な手法であり，１パスで高速にテクスチャ画像を生成可能である．

しかし，これらのノンパラメトリックなテクスチャ合成法では貪欲法により画素値あるいは画像パッチを選択するため，見た目が不自然となる不連続箇所が発生する問題があった．これを避けるためには，画像全体で矛盾が生じないように画素を選択してテクスチャ画像を合成

する必要がある．これを実現する手段として，Kwatra らは EM アルゴリズムに似た計算方法で全体最適化を試みている [2]．この方法は貪欲法により選択した画像パッチを張り合わせ，画像パッチが重複する箇所は画像パッチの平均画素値で置き換える操作を繰り返すことで全体最適化を図っている．一方で，決定論的ではなく確率論的なテクスチャ合成法が Paget らに提案されている [4]．この方法では学習画像の画像パッチの画素数を次元数とした多次元ヒストグラムを作成し，入力画像パッチに対する画素値を前記多次元ヒストグラム分布から確率的に決定する．この手法と多重解像度を組み合わせることで自然物のテクスチャ合成を違和感なく生成することができるとの報告がある．しかしながら，多次元ヒストグラムに必要なメモリは画像パッチの画素数に指数関数的に爆発する．これに対して Paget らは大規模なワークステーションを用いて計算を行っている．

これに対して，本論文では新しいノンパラメトリックな画素貼り付け型の確率論的なテクスチャ合成モデルを提案する．本手法は Paget の方法 [4] のように大量のメモリを必要とせず，全体最適化により画素値を選択するため自然な合成画像が通常のＰＣ環境にて計算可能である．また，従来の WL 法 [1] との関連を示し，提案モデルが WL 法を含む，より広いモデルであることを示す．提案手法は高品質のテクスチャ画像が合成可能であるが，WL 法と比べて多くの処理時間を要する．そこで，提案モデルの式より処理時間のボトルネックとなる箇所を抽出し，この部分を効果的に近似する方法を提案する．

テクスチャ画像の合成実験では，提案手法が従来の WL 法よりも画像の連続性が保たれた自然なテクスチャ合成が可能なことを示す．さらに，提案する近似手法が近似を行わない場合と比べても結果に遜色がなく，一方で処理時間が約 5 倍程度高速化できることを示す．

2. テクスチャ合成モデル

本章では，まず使用する画像モデルについて簡単に述べ，提案するテクスチャ合成モデルについて述べる．

2.1 マルコフ確率場による画像のモデル化 [12]

画像データを $\mathbf{x} = \{x_1, x_2, ..., x_H\}$ と表し，各画素 x_s は画素値 $\lambda \in \Lambda$ を取る．Λ は取りうる画素値の集合で H は全画素数である．ここで，各画素 x_s が $N \times N$ 近傍の画素集合 $\mathbf{x}_{\mathcal{N}_s}$ (x_s を含まない) に依存して画素値が確率的に決定される，マルコフ確率場モデルを本手法で採用する．画像の全画素の同時確率 $P(\mathbf{x})$ を次のように各画素の確率 $P\left(x_s|\mathbf{x}_{\mathcal{N}_s}\right)$ の積として近似する．

$$P\left(\mathbf{x}\right) \approx \prod P\left(x_s|\mathbf{x}_{\mathcal{N}_s}\right). \tag{1}$$

確率分布 $P(\mathbf{x})$ はギブスサンプラを用いて推定することができる．すなわち，適当に与えた初期値 $\mathbf{x}^{(0)}$ から開始して，$\mathbf{x}^{(t)}$ での確率分布に従って $\mathbf{x}^{(t)}$ から 1 画素を更新した $\mathbf{x}^{(t+1)}$ を次々と抽出していく．$t \to \infty$ のとき，$\mathbf{x}^{(t)}$ の分布は $P(\mathbf{x})$ に収束することが知られている．実際には所定の回数 $t = T_1$ で計算を行い，初期値の影響が十分小さくなった $\mathbf{x}^{T_0}, \mathbf{x}^{T_0+1}, ..., \mathbf{x}^{T_1}$ の平均値を最終的な推定値として採用する．

2.2 テクスチャ合成モデル

提案モデルは学習画像 Y を入力として，テクスチャ画像 X を合成する．各画素の画素値は次式で定義される確率 $P\left(x_s|\mathbf{x}_{\mathcal{N}_s}, Y\right)$ に従って生起するとする．

$$P\left(x_s|\mathbf{x}_{\mathcal{N}_s}, Y\right) \equiv \frac{p\left(x_s|\mathbf{x}_{\mathcal{N}_s}, Y\right)}{\sum_{\lambda\in\Lambda} p\left(\lambda|\mathbf{x}_{\mathcal{N}_s}, Y\right)} \tag{2}$$

ここで，尤度 $p\left(x_s|\mathbf{x}_{\mathcal{N}_s}, Y\right)$ は次式で定義する．

$$p\left(x_s|\mathbf{x}_{\mathcal{N}_s}, Y\right) \equiv K^{(0,0)}(\mathbf{x}_s, Y), \tag{3}$$

ここで，$\mathbf{x}_s$ は図 1(a) に示すような x_s の $N \times N$ 近傍の画素集合 (x_s 含む．以降，パッチと呼ぶ) である．カーネル関数 K は次式で定義される．

$$\begin{aligned} K^{\boldsymbol{v}}(\mathbf{x_s}) &\equiv \sum_{k\in Y} \exp\left(-\frac{\|\mathbf{x}_s^{(\boldsymbol{v})} - \mathbf{y}_k^{(\boldsymbol{v})}\|}{\sigma}\right) \\ &\qquad \times \exp\left(-\frac{x_{s+\boldsymbol{v}} - y_{k+\boldsymbol{v}}}{\sigma}\right) \qquad (4) \\ &\equiv \sum_{k\in Y} w_k \exp\left(-\frac{x_{s+\boldsymbol{v}} - y_{k+\boldsymbol{v}}}{\sigma}\right). \qquad (5) \end{aligned}$$

ここで $\boldsymbol{v}$ は画素 x_s からの相対的な位置ベクトルを示し，$\mathbf{x}_s^{(\boldsymbol{v})}$ は $\mathbf{x}_s$ から $x_{s+\boldsymbol{v}}$ を取り除いた画素集合である（図 1(b) 参照）．σ はカーネルのパラメタである．

式 (4) は出力画像のパッチ $\mathbf{x}_s$ と学習画像の全パッチ $\mathbf{y}_s$ との二乗誤差が小さいほど大きな値を返すような画像パッチ間の類似度を評価する関数である．これは式 (5) のように考えることもできる．すなわち，注目画素 $x_{s+\boldsymbol{v}}$ と学習画像との類似度は，近傍画素が良く似ている学習画像パッチを優先的に選択（式では近傍画素の類似度を w_k と表記）することとなる．

次に，$\mathbf{x}^{(t)}$ から 1 画素 x_s を更新して $\mathbf{x}^{(t+1)}$ を得る際の，x_s の画素値の選択確率について考える．このとき，同時確率は x_s を近傍に持つ画素集合のみで表すことができる．

$$\begin{aligned} P(x_s|\mathbf{x}_{(s)}, Y) &= \prod_{x_s\in\mathbf{x}} P(x_s|\mathbf{x}_{(s)}, Y) \qquad (6) \\ &\propto \prod_{x_s\in\mathbf{x}_{\mathcal{N}_s}} P(x_s|\mathbf{x}_{\mathcal{N}_s}, Y). \end{aligned}$$

$\mathbf{x}_{(s)}$ は x_s を除く全画素集合である．式 7 より x_s の画素値が変化すると，x_s の周囲の画素の尤度も変化する．周囲の尤度変化を考慮した x_s の結合尤度関数は次式となる．

$$p(x_s|\mathbf{x}_{(s)}, Y) = \frac{1}{Z} \prod_{\boldsymbol{v} \in \mathcal{N}_{offset}} K^{\boldsymbol{v}}(\mathbf{x}_{s+\boldsymbol{v}}) \qquad (7)$$

ここで，Z は正規化定数で，$\mathcal{N}_{offset}$ は x_s から近傍画素への位置ベクトルを示す．例えば，図 1(a) の場合は $\mathcal{N}_{offset} = \{(-1,-1),(0,-1),(1,-1),\cdots(1,1)\}$ の 9 つの要素となる．上式の結合尤度関数に従ってサンプリングを繰り返し，得られたサンプル列の平均を取ることで提案する確率的なテクスチャ画像合成が実現する．

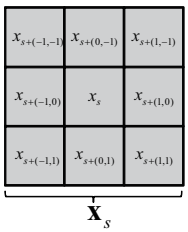
2.3 K 近傍サンプリングによる高速化

式 (5) をみると，学習画像 Y の画素数 M に比例して処理時間が多くなることが分かる．学習画像のサイズが大きな場合，これが処理のボトルネックとなる．しかし，式 (5) のカーネル関数は入力画像パッチとの 2 乗差 d^2 に対して指数的に急減少するため，入力画像パッチとの類似度が小さな学習画像パッチは尤度関数への寄与が少ない．そこで，入力画像パッチと大きく異なる学習画像パッチを取り除くことを考える．具体的には，入力画像パッチに対してユークリッドノルムが小さな順に K 個の学習画像パッチを取り出して（以下,K 近傍サンプルと呼ぶ）, これのみを式 (5) の計算に用いる．すなわち，式 (5) を以下のように近似する．

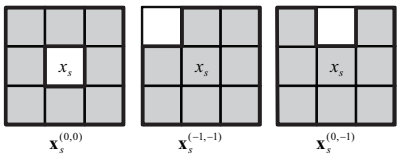
$$K^{\boldsymbol{v}}(\mathbf{x_s}) \approx \sum_{k \in \mathcal{N}^K_{Y,\mathbf{x}_s}} w_k \exp\left(-\frac{x_{s+\boldsymbol{v}} - y_{k+\boldsymbol{v}}}{\sigma}\right). \qquad (8)$$

ここで，$\mathcal{N}^K_{Y,\mathbf{x}_s}$ は入力画像パッチ $\mathbf{x}_s$ と Y との K 近傍サンプルの集合を示す．

M 個のサンプルから K 近傍のサンプルを効率よく取り出すアルゴリズムおよびデータ構造として Kd 木を用いることができる．Kd 木は最良で $\mathcal{O}(\log N)$ のオーダで K近傍サンプルを探索可能である．Kd 木による近傍サンプルの探索はコンピュータグラフィクス分野におけるフォトンマップ法などで大きな成果を得ている [15]．一般に Kd 木探索において探索すべきベクトルの次元数が増えると, 探索効率が著しく下がり全探索と計算量が変わらなくなるということが指摘されている [16]．そこで, 本研究では厳密な近傍サンプルは取得できないものの高次元ベクトルにおいても高速に探索可能な近似近傍探索 (ANN) [16] を組みこんだ Kd 木を用いる．



(a) 近傍画素集合



(b) 1 点を取り除いた近傍画素集合

図 1　近傍画素集合の定義
Fig. 1　Definitaion of the neghbor pixels set

2.4 従来モデルとの関連

従来の WL 法は，式 (5) のカーメルパラメタ $\sigma \to 0$ として，さらに式 (7) の結合尤度の計算において $\mathcal{N}_{offset} = \{(0,0)\}$ とした場合に相当する．すなわち，カーネルパラメタ σ を小さくすると入力画像パッチ $\mathbf{x}_s$ に最も類似する学習画像パッチ $\mathbf{y}_k$ の重み w_k が他の学習画像パッチの重み $w_i (i \neq k)$ に比べて相対的に極端に大きくなる（$w_k = 1, w_i = 0 (i \neq k)$ となる）．また，WL 法は注目画素 x_s を更新したときに周囲の画素への影響を考えていないため，$\mathcal{N}_{offset} = \{(0,0)\}$ となる．

なお，周囲の画素への影響を考えた決定論的な k-コヒーレント・テクスチャ合成法 [14] も提案されているが, これは提案モデルで $\sigma \to 0$ とした場合に相当する．このように提案モデルは従来の決定論的なテクスチャモデルを包含する形になっている．

ただし，両者のケースにおいて，決定論的な方法は 1 パスでテクスチャ画像を生成できるように近傍画素のマスク形状が若干異なる．

3. テクスチャ合成アルゴリズム

提案モデル式を用いたテクスチャ合成アルゴリズムについて述べる．本アルゴリズムの全フローを図 2 に示す．各ステップについて以下に述べる．

3.1 画像ピラミッド画像生成ステップ

WL 法と同様に学習入力画像 Y に対して L_{max} 段階の画像ピラミッドを生成し，低解像度画像 ($L = L_{max}$) から原画像 (L=1) の順にテクスチャ画像を合成する．L 段階のテクスチャ画像合成においては，それまで合成した低解像度のテクスチャ画像 $(L+1, L+2, \cdots L_{max})$ を入力画像パッチ $\mathbf{x}_s$ に含ませる．例えば，図 3(a) では同図の白丸で示す $\mathbf{x}_s = (x_1^L, x_2^L, \cdots x_{25}^L, x_1^{L+1}, x_2^{L+1}, \cdots, x_9^{L+1})$ が画像パッチとなる．このような多重解像処理を組み込むことにより，大局的なテクスチャおよび局所的なテクスチャの両方の特徴をとらえつつ合成することができる．

本アルゴリズムにおいては画像ピラミッドは lanczos 縮小フィルタ [17] を用いて生成する．また，L 段階の画素 (x, y) に対して，$L+1$ 段階の画素値を参照する際，従来手法では $(\lfloor x/2 \rfloor, \lfloor y/2 \rfloor)$ と整数値に打ち切って画素値

を参照していた (図 3(a)) [13]．しかし，この方法では出力画像にシャギーが発生する傾向が見られたため，本アルゴリズムでは $(x/2, y/2)$ として lanczos 補間により画素値を参照する (図 3(b))．

3.2 テクスチャ画像合成ステップ

現在処理中の多重解像度の段階を L とする．出力画像を $\mathbf{x}^{L,t=0}, \mathbf{x}^{L,t=1}, ..., \mathbf{x}^{L,t=T_1}$ と更新していき，画像

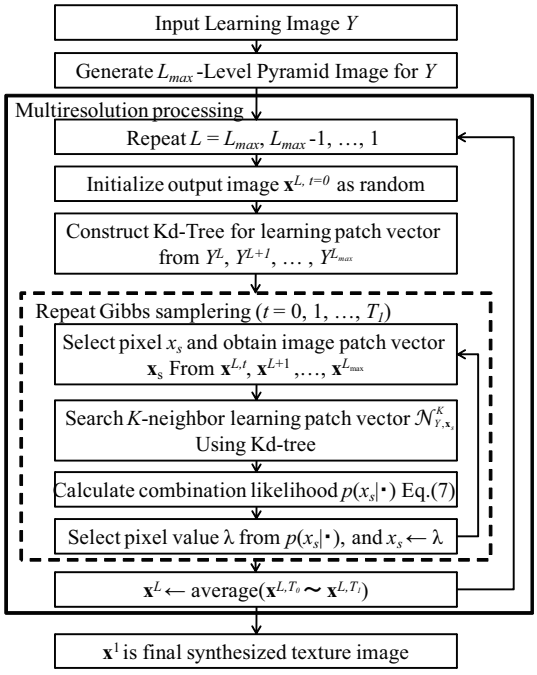
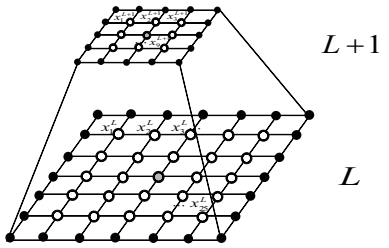
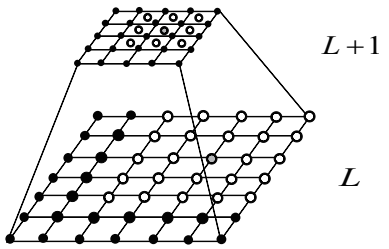


図 2　提案手法のフロー
Fig. 2　A flowchart of the proposed method



(a) 整数位置の画素参照



(b) 実数位置の画素参照

図 3　画像ピラミッドにおける画素参照方法
Fig. 3　A feature vector for the pyramid image

$\mathbf{x}^{L,t=T_0}, ..., \mathbf{x}^{L,t=T_1}$ を平均化することで L 段階のテクスチャ画像を合成する．具体的には以下のステップにより合成する．

STEP_1　$L = L_{max}$ とする．

STEP_2　出力画像 $\mathbf{x}^{L,t=0}$ の各画素値をランダムに初期化する．具体的には，学習画像の画素をランダムに取り出し，出力画像 $\mathbf{x}^{L,t=0}$ にコピーする．

STEP_3　学習画像 Y^L の各画素 $y_s \in Y^L$ を中心とする画像パッチ $\{\mathbf{y}_s\}$ を，$Y^L, Y^{L+1}, \cdots, Y^{L_{max}}$ から生成する．そして，任意のパッチ $\mathbf{x}_s$ に近い $\mathbf{y}_s$ を高速に探索できるように，$\{\mathbf{y}_s\}$ に関する Kd 木の構築を行う．

STEP_4　画像の左上から右下へラスタスキャンで更新対象の注目画素 x_s を選択し，x_s に対する画像パッチ $\mathbf{x}_s$ を作成する．次に，Kd 木を用いて $\mathbf{x}_s$ とのユークリッド距離が近い K 個の学習パッチ集合 $\mathcal{N}^K_{Y,x_s}$ を得る．そして，式 (5) を用いて各画素値 $\lambda \in \Lambda$ に対する結合尤度を求める．乱数を発生させ，結合尤度の確率分布に従って画素値 λ を得て，注目画素値 $x_s \leftarrow \lambda$ と更新する．

STEP_5　$t \leftarrow t+1$ として，$t = T_1$ となるまで STEP_3～STEP_4 を繰り返す．

STEP_6　$L \leftarrow L-1$ として，$L = 1$ となるまで STEP_2～STEP_5 を繰り返す．

上記ステップを繰り返して最終的に得られる $\mathbf{x}^{L=1}$ が合成したテクスチャ画像である．

4. 実 験 結 果

サイズ 128×128 の 32 階調のグレースケール画像を学習画像としてサイズ 256×256 の画像を合成した．用いた計算機は CPU Intel Core-i7 2.8GHz およびメモリ 16GByte で，実行プログラムは C++で実装した．

比較手法として，提案手法と同様にピクセルベースのノンパラメトリックテクスチャ合成法である WL 法を用いた．WL 法はラスタ走査した処理画素に対して，L 字型の探索ウィンドウを用いて学習サンプルとの 2 乗差が最小となる画素を探索して，その中心画素値を処理画素に一意に割り当てる方法である．両者の方法において多重解像度処理 ($L_{max} = 3$) を用い，近傍画素の幅 $N = 7(7 \times 7$ の画像パッチ) を用いた．提案手法では $\sigma = 0.5$，T_0, T_1 は出力画像の画素数を H として，$T_0 = 3H, T_1 = 25H$ と設定した．また，K 近傍サンプリングのパラメタは $K = 100$ を用いた．図 4(a)～図 4(c) は多重解像度によるテクスチャ画像の合成過程を示す．

4.1 合成画像結果に関する考察

図 5(a)～図 7(a) に対する合成結果をそれぞれ図 5(b)，図 5(c)～図 7(b)，図 7(c) に示す．図 5(a) に関しては WL 法も提案手法も似たような結果が得られた．一方で，図 6(a) および図 7(a) に関しては目視で違いが分かるほどの差が見られた．図 5(d)，図 5(e)～図 7(d)，図 7(e) に一部の拡大図を示す．

WL 法は 1 パスで高速に合成できるものの, 貪欲法により一意に画素を割り当てるため局所解が発生し，ところどころ目視で確認できるほどの不自然な不連続箇所が発生した．特に大きな違いが見られた図 6(a) において，合成画像の葉の形状に関して，WL 法ではうまく再現できていないが，提案手法では入力学習画像と遜色なく再現できた．これは，提案手法は画素間で矛盾が生じないように全体最適化を行って画素値を推定するため WL 法のような局所解が発生しにくく，不連続箇所がない自然な画像が得ることができたと考えている．

4.2 K 近傍サンプリングに関する定量評価

提案する K 近傍サンプリングの効果を確かめるために，$K = 10, 30, 50, 100, 300$ それぞれの値で K 近傍サンプリングを行った場合と，近似なしの場合（式 (5) を使用）の結果を比較した．入力学習画像は図 6(a) として，サイズ 128×128 のテクスチャ画像を合成した．

提案手法は確率的に画像を合成するため，初期値や入力パラメタが異なると出力画像も大きく変化し比較評価が難しくなる．そこで $K = 10, 30, 50, 100, 300$ 全ての場合において，$L = 3$ の段階では近似なしの結果を用いて，それ以降の段階では，各条件での画像合成処理を実行した．それぞれの結果を図 8(a)〜図 8(f) に示す．また，図 9(a)〜図 9(c) に $K = 10, 100$ および近似なしの場合の一部拡大図を示す．同図より K の値を小さくしすぎると画像の一部がぼけたような出力が得られるため，ある程度の K の値を用いる必要があることが分かる．

このことについて，定量的な評価を行うために各条件での近似なしの場合との画像誤差および処理時間を評価した．この結果を表 1 および図 10(a) および図 10(b) に示す．ここで 2 枚の画像 $f(x, y), g(x, y)$ の誤差は次式の平均二乗誤差 (MSE) および PSNR で評価した．

$$\mathrm{MSE} = \frac{1}{H} \sum_{x,y} \left(f(x, y) - g(x, y) \right)^2 \tag{9}$$

$$\mathrm{PSNR} = \log_{10} \left(\frac{255}{\mathrm{MSE}} \right) \tag{10}$$

また，処理時間を含めた性能を PSNR/処理時間 で評価した．

表より，$K = 300$ の場合は画像の誤差が最も少ないものの処理時間が大きくなり，全体の評価としては悪くなってしまった．一方で $K = 100$ の場合は，精度および処理時間ともにバランスが取れた結果となった．$K = 100$ の場合は 6.83[h] となっており，近似なしの場合の処理時間 35.50[h] と比べて約 5 倍の処理時間の短縮が実現できた．

表 1　計算時間および画像の誤差の比較
Table 1　Comparison of the calculation time

K	MSE	PSNR	Time[h]	$\frac{\mathrm{PSNR}}{\mathrm{Time}}$
10	58.71	0.64	4.71	0.14
30	37.58	0.84	5.38	0.16
50	24.09	1.02	5.80	0.18
100	11.09	1.36	6.83	0.20
300	4.30	1.77	15.15	0.12
No approx.	-	-	35.50	-

5. むすび

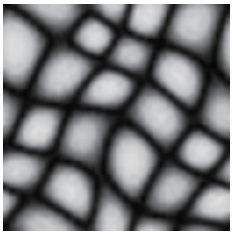
確率的ノンパラメトリックなテクスチャ画像合成モデルおよびテクスチャ画像合成アルゴリズムを提案した．提案手法はギブスサンプラを用いて出力画像を反復的に更新することで，従来の貪欲法的なテクスチャ画像合成手法に比べて局所解に陥りにくく自然な結果が得られる．また，提案モデル式より，処理時間のボトルネックが学習画像の画素サイズに依存する点を指摘し，この部分を Kd 木を用いた K 近傍探索によって効果的に近似する方法を提案した．

実験では，従来の WL 法と比べて目視で分かるほどの不自然な領域が発生せず，学習画像に忠実なテクスチャ画像を合成できることを示した．また，提案した近似モデルが近似しない場合と比べて，同等の品質が得られ処理時間が約 5 倍高速化できることを示した．

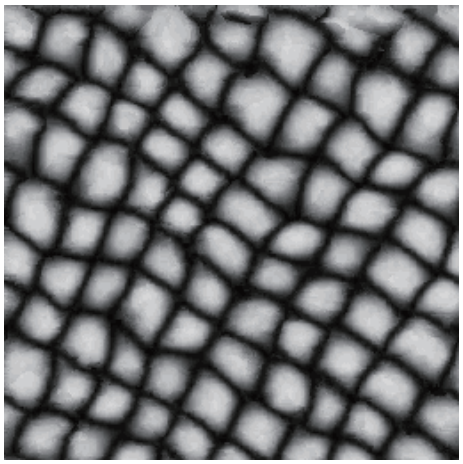
今後の展望として，本モデルを用いたテクスチャ合成技術の応用先の模索，および本モデルを用いたテクスチャを含む物体認識や領域分割法を検討する．

(a) level=3　(b) level=2　(c) level=1

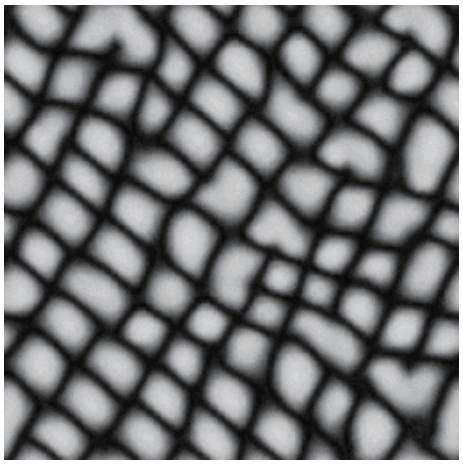
図 4　多重解像度による合成
Fig. 4　An example of the multi-resolution image synthesis



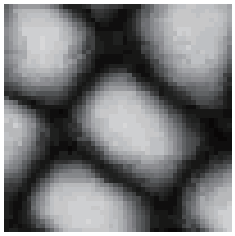
(a) 入力画像



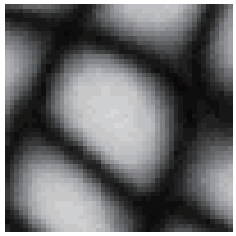
(b) WL 法による合成結果



(c) 提案手法による合成結果



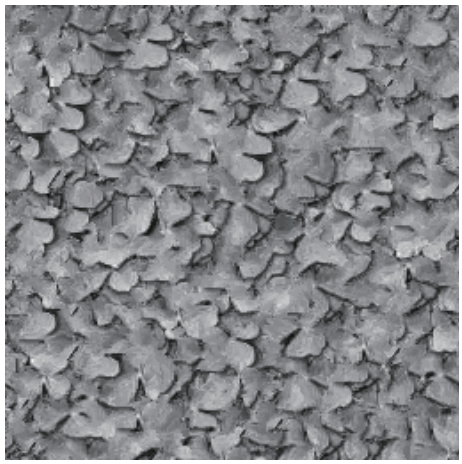
(d) 拡大 (WL 法)



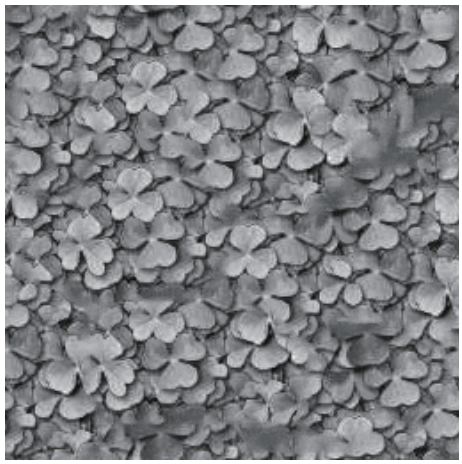
(e) 拡大 (提案手法)

図 5　合成結果の比較 1
Fig. 5　Results of synthesized image

(a) 入力画像

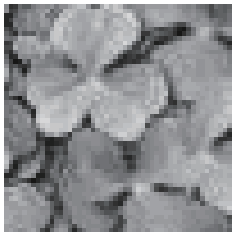


(b) WL 法による合成結果



(c) 提案手法による合成結果

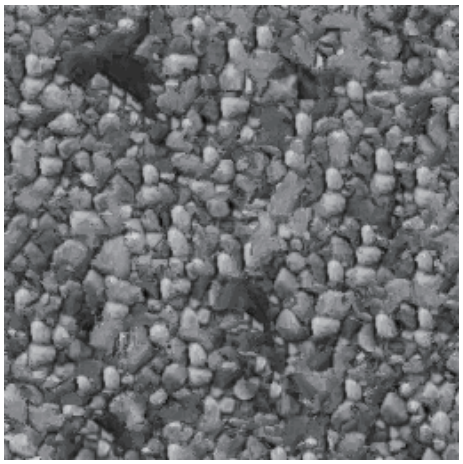
(d) 拡大 (WL 法)



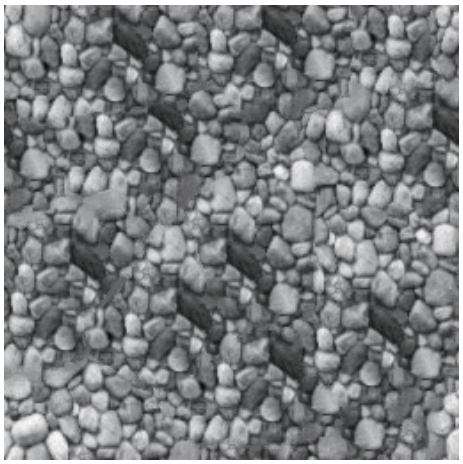
(e) 拡大 (提案手法)

図 6　合成結果の比較 2
Fig. 6　Results of synthesized image

(a) 入力画像

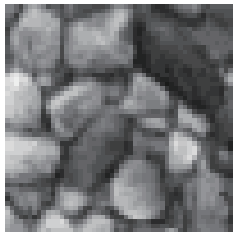


(b) WL 法による合成結果



(c) 提案手法による合成結果

(d) 拡大 (WL 法)



(e) 拡大 (提案手法)

図 7　合成結果の比較 3
Fig. 7　Results of synthesized image

(a) $K = 10$

(b) $K = 30$

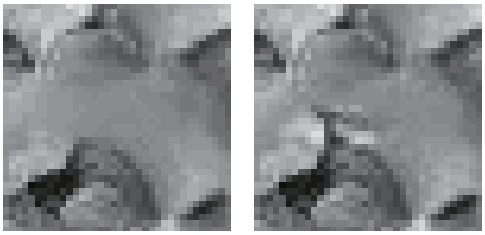
(c) $K = 50$

(d) $K = 100$

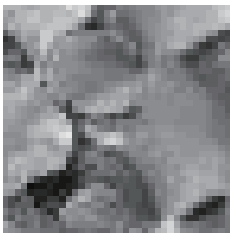
(e) $K = 300$

(f) 近似なし

図 8　合成結果の比較
Fig. 8　Results

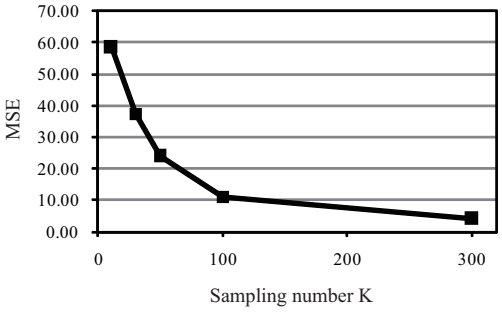


(a) $K = 10$ (b) $K = 100$

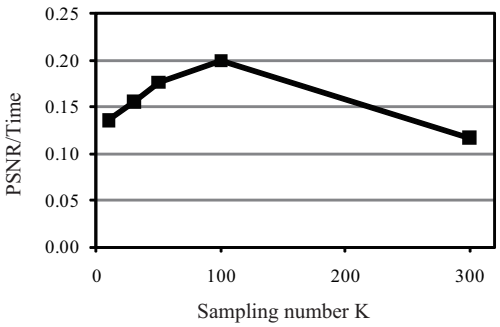


(c) 近似なし

図 9 拡 大 図
Fig. 9 Zoomed results



(a) MSE との関係



(b) $\frac{\mathrm{PSNR}}{\mathrm{Time}}$ との関係

図 10 K と画像誤差の関係
Fig. 10 Relationship between K and image errors

文 献

[1] L.Wei and M.Levoy, "Fast texture synthesis using tree-structured vector quantization," in Proceedings of the 27th annual conference on Computer graphics and interactive techniques, SIGGRAPH '00, pp.479-488, no.10, 2000

[2] V.Kwatra, I.Essa, A.Bobick and N.Kwatra, "Texture optimiation for example-based synthesis", SIGGRAPH'05, 2005

[3] L.Wei, L.Sylvain, K.Vivek, and T.Greg, "State of the Art in Example-based Texture Synthesis," in Eurographics '09 State of the Art Reports(STARs), Eurographics, 2009

[4] R.Paget and I.D.Longstaff, "Texture synthesis via a noncausal nonparametric multiscale Markov random field," IEEE Transactions on Image Processing, vol.7, no.6, pp.925-931, 1998.

[5] 森俊二, 坂倉栂子, "画像認識の基礎 (2) -特徴抽出・エッジ検出・テクスチャー解析-," オーム社, 1990.

[6] S.Geman, D.Geman, "Stochastic Relaxation, Gibbs Distributions, and the Bayesian Restoration of Images," IEEE Transactions on Pattern Anal. Mach. Intell., vol.6, no.6, pp.721-741, 1984.

[7] R.George and K.Anil, "Markov random field texture models," IEEE Transactions on Pattern Anal. Mach. Intell., vol.5, no.1, pp.25-39, 1983.

[8] 伊庭幸人, "ベイズ統計と統計物理," 岩波講座 物理の世界, 2003.

[9] R.Chellapa, "Two-dimensional discrete gaussian markov random field models for image processing," Pattern Recognition 2, pp.79-112, 1985.

[10] 田中 和之, "確率モデルによる画像処理技術入門," 森北出版, 2006

[11] A.Efros and W.T. Freeman, "Image quilting for texture synthesis and transfer," in Proceedings of the 28th annual conference on Computer graphics and interactive techniques, SIGGRAPH '01, pp.341-346, 2001.

[12] 青木工太, 長橋宏, "マルコフ確率場と階層的事前分布による画像分割," 画像電子学会誌, vol.35, no.4, pp.286-295, 2006.

[13] J.S.D.Bonet, "Multiresolution sampling procedure for analysis and synthesis of texture images," In Proceedings of SIGGRAPH'97, pp.361-368, 1997.

[14] J.Han, K.Zhou, Li-Yi Wei, M.Gong, H.Bao, X. Zhang, and B.Guo, "Fast example-based surface texture synthesis via discrete optimization," Vis. Comput., vol.22, no.9, pp.918-925, 2006.

[15] J.Henrik, 苗村 健, "フォトンマッピング—実写に迫るコンピュータグラフィックス," オーム社, 2002

[16] S.Arya, D.M.Mount, N.S.Netanyahu, R.Silverman, and A.Y.Wu, "An optimal algorithm for approximate nearest neighbor searching in fixed dimensions," Journal of the ACM, vol.45, No.6, 1998.

[17] C.E.Duchon,"Lanczos filtering in one and two dimensions," Journal of Applied Meteorology, vol.18, no.8, pp.1016-1022, 1979