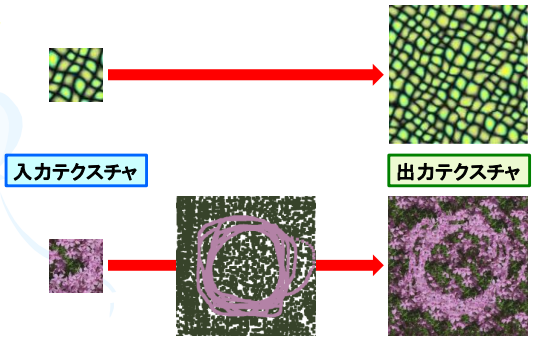
イメージシンセシス

テクスチャ合成
（ピクセルベース手法）

岩手大学大学院
総合科学研究科 理工学専攻
デザイン・メディアエ学コース
藤本　忠博

テクスチャ合成

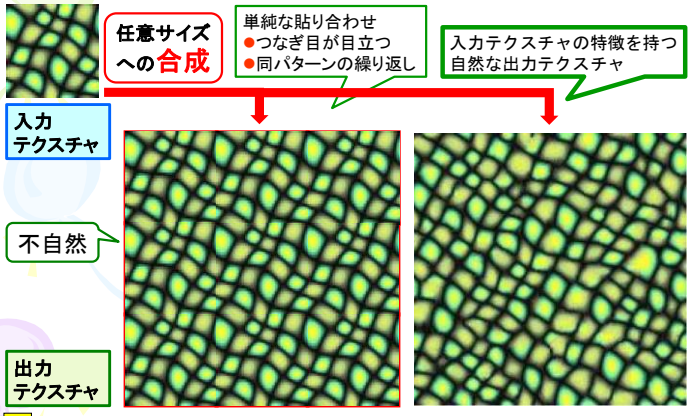
■ 入力テクスチャからサイズの異なる自然な出力テクスチャを合成する.



L. Wei, M. Levoy, “Fast Texture Synthesis using Tree-structured Vector Quantization”, 2000

M. Ashikhmin, “Synthesizing Natural Textures”, 2001

テクスチャ合成

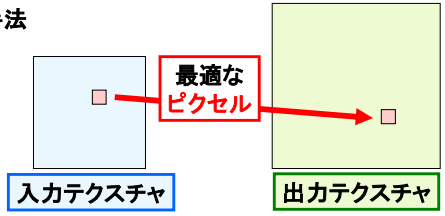


L. Wei, M. Levoy, “Fast Texture Synthesis using Tree-structured Vector Quantization”, 2000

テクスチャ合成の分類

■ ピクセルベース手法

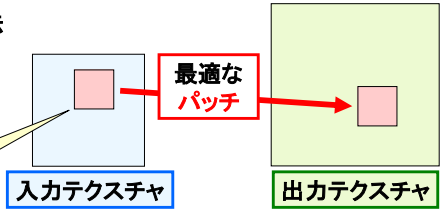
入力テクスチャから適切なピクセルを選択し，そのピクセルの色を出力テクスチャに割り当てる.



■ パッチベース手法

入力テクスチャから部分テクスチャ（パッチ）を切り出し，つなぎ目が目立たないように貼り合わせる.

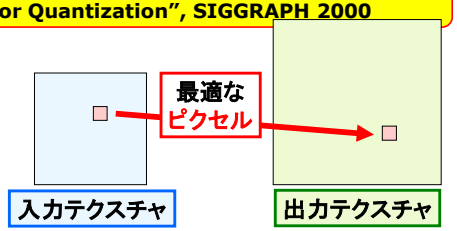
一つのパッチは幾つかのピクセルからなる.



ピクセルベース手法

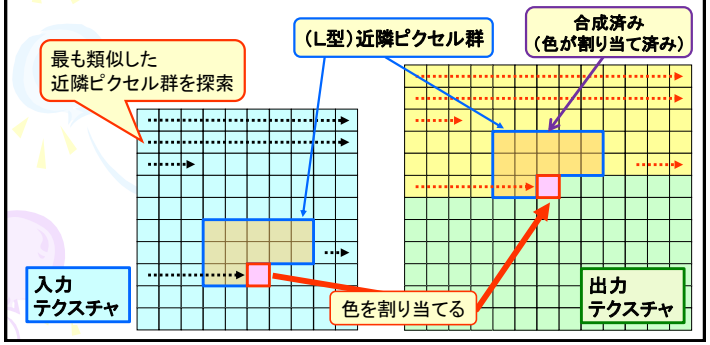
- 入力テクスチャから適切なピクセルを選択し，そのピクセルの色を出力テクスチャに割り当てる．
- **A. A. Efros, T. K. Leung (1999), L. Wei, M. Levoy (2000)** により基本アルゴリズムが提案され，その後，様々な手法が提案されている．
- **A. A. Efros, T. K. Leung, "Texture Synthesis by Non-parametric Sampling", 1999**
- **L. Wei, M. Levoy, "Fast Texture Synthesis using Tree-structured Vector Quantization", SIGGRAPH 2000**

これ以降，
この論文の内容に
基づいて説明する．



ピクセルベース手法の基本アルゴリズム

- 入力テクスチャから選択したピクセルの色を出力テクスチャのピクセルに**走査線順**に割り当てていく．
- 出力テクスチャ上の各ピクセルについて，**(L型)近隣ピクセル群**の色が最も類似したピクセルを入力テクスチャから選ぶ．

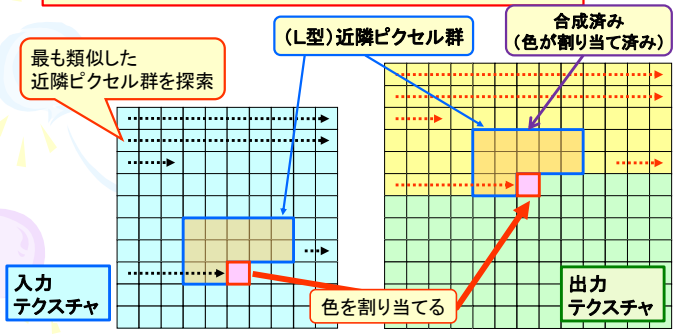


ピクセルベース手法の基本アルゴリズム

- 入力・出力テクスチャの近隣ピクセル群の類似度の評価関数

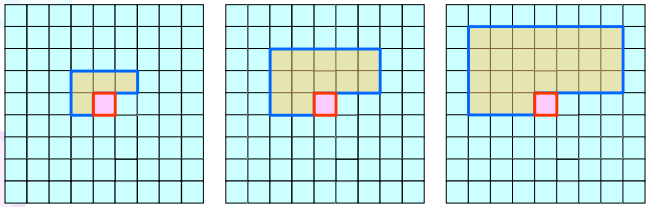
$$D=\sum_p\left\{(R_I(p)-R_O(p))^2+(G_I(p)-G_O(p))^2+(B_I(p)-B_O(p))^2\right\}$$

p：近隣ピクセル群のピクセル



近隣ピクセル群

- 近隣ピクセル群のサイズ
  - サイズが大きなほど → 合成結果は良好 → 計算時間は増加
  - 良好な合成結果を得るためには，入力テクスチャの特徴的パターンの構造よりも大きなサイズが必要である．
  - **Wei, Levoy**の論文中には上記の記述があるが，実際には良好な合成結果を得る最適なサイズは特徴的パターンの構造に依存し，必ずしも大きいほど良いわけではない． → **Ashikhmin**の論文



3×3
(実際は3×2)

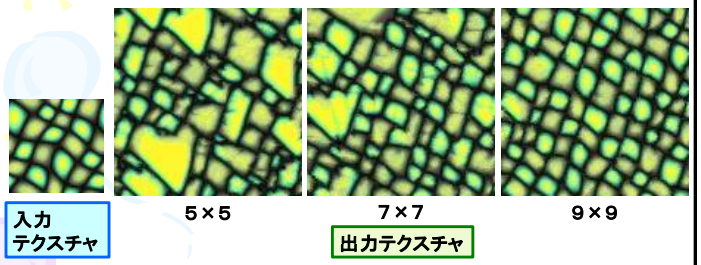
5×5
(実際は5×3)

7×7
(実際は7×4)

近隣ピクセル群

- 近隣ピクセル群のサイズを変えた実験結果

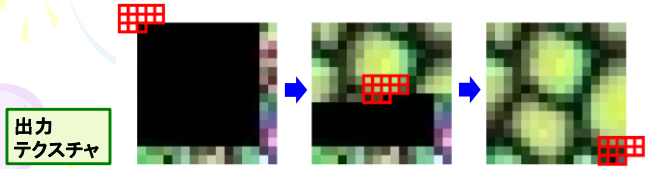
出力テクスチャのサイズ： 128×128



L. Wei, M. Levoy, "Fast Texture Synthesis using Tree-structured Vector Quantization", 2000

出力テクスチャの初期化と環状処理

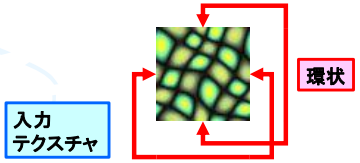
- 出力テクスチャの各ピクセルの色を**ランダムノイズ（ノイズ色）**で**初期化**.
- 出力テクスチャの**境界**を**環状（上端と下端，左端と右端が連続）**に扱う.
- 出力テクスチャのピクセルを**走査線順**に処理
  - 上端と左端の数ピクセル幅（近隣ピクセル群のサイズで決定）に含まれるピクセルについては，合成の際，その近隣ピクセル群には，合成済みの色ではなく，初期化によるノイズ色が含まれる.
  - 実際には，下端と右端の数ピクセル幅のノイズ色だけが使われる.



L. Wei, M. Levoy, "Fast Texture Synthesis using Tree-structured Vector Quantization", 2000

入力テクスチャの環状処理

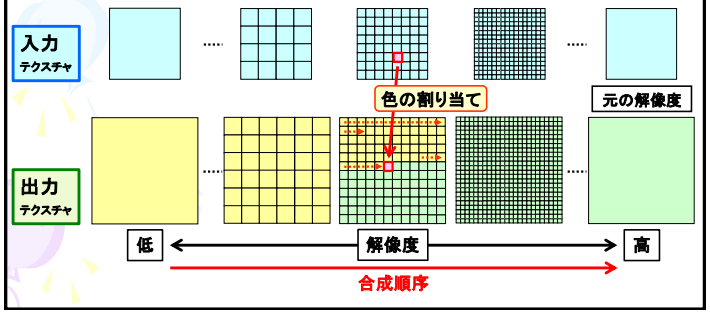
- 入力テクスチャの**境界**が**環状（上端と下端，左端と右端が連続）**かどうかにより，出力テクスチャの各ピクセルに対する近隣ピクセル群の類似度評価による入力テクスチャのピクセルの探索を次のように行う.
  - **環状の場合**
    全てのピクセルを探索する．境界部分のピクセルに対する近隣ピクセル群は境界を越えて反対側に含まれるが，その内部のテクスチャは連続している.
  - **環状でない場合**
    近隣ピクセル群が境界を超えないピクセルだけを探索する.



L. Wei, M. Levoy, "Fast Texture Synthesis using Tree-structured Vector Quantization", 2000

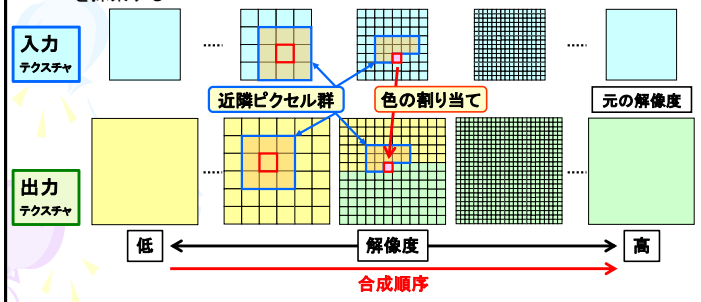
多重解像度処理

- 入力・出力テクスチャの解像度を階層的に下げて**多重解像度化**する.
- **低解像度から高解像度の順**で出力テクスチャを合成していく.
- 各解像度レベルの合成では，単一解像度の場合と同様，出力テクスチャのピクセルを**走査線順**で処理し，同レベルの入力テクスチャのピクセルの色を割り当てていく.



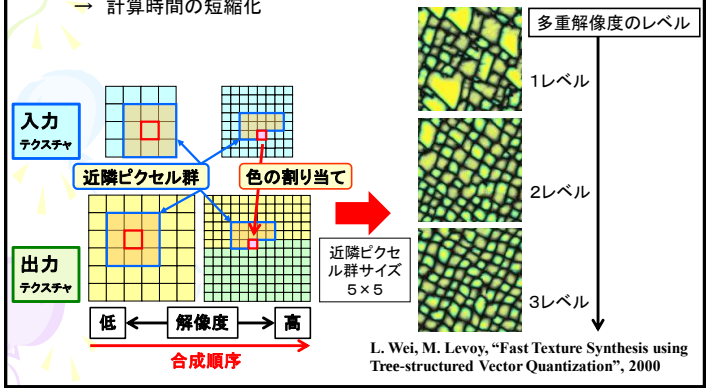
多重解像度処理

- 類似度を評価する**近隣ピクセル群**は，**複数の解像度レベル**で構成する．
  - **処理中のレベル**： 単一解像度の場合と同じ**L型領域N_L**
  - **より低解像度のレベル**： **領域N_L**を含む**正方領域**（合成済み）
- 単一解像度の合成と同じ評価関数を用い，最も類似した近隣ピクセル群を探索する．



多重解像度処理

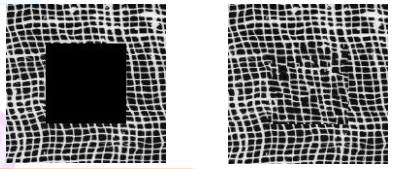
- 多重解像度処理により，近隣ピクセル群のサイズが小さな場合でも，単一解像度でサイズが大きな場合に匹敵する良好な合成結果が得られる．
  → 計算時間の短縮化



L. Wei, M. Levoy, “Fast Texture Synthesis using Tree-structured Vector Quantization”, 2000

制約付きテクスチャ合成

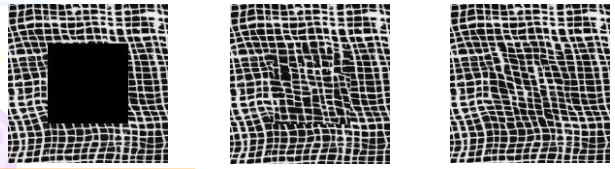
- テクスチャの**欠落した領域**だけを合成する．
  - 周囲と類似のパターンにする．
  - 合成した領域の境界が目立たないようにする．
- 欠落していない領域を入力テクスチャとして用い，通常の方法で欠落領域を合成した場合，欠落領域の下端と右端の境界で**不連続**となる．
  - **L型領域（非対称領域）**の**近隣ピクセル群**による問題
  - **ピクセルの走査線順の処理**による問題



L. Wei, M. Levoy, “Fast Texture Synthesis using Tree-structured Vector Quantization”, 2000

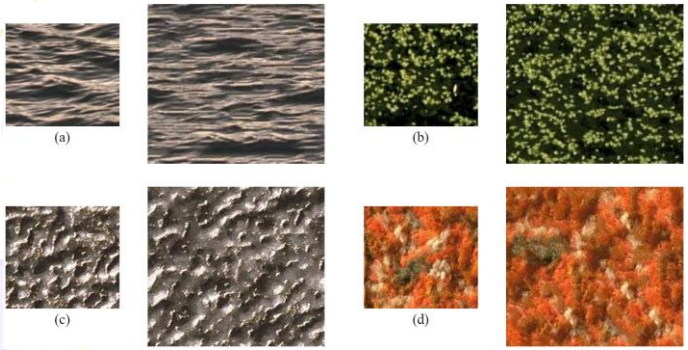
制約付きテクスチャ合成

- **解決策1**： **多重解像度処理**で，各解像度レベルの合成を**2パス**で行う．各パスで，**近隣ピクセル群**の構成を変え，合成済みピクセルからなる**正方領域（対称領域）**から構成されるようにする．
  - **1パス目**：処理中レベルより低解像度レベルの正方領域だけを使う．
  - **2パス目**：処理中レベルの正方領域も含める．
- **解決策2**： **ピクセルの処理順序**を変える．
  - 欠落領域の境界から内側に向かって合成する．



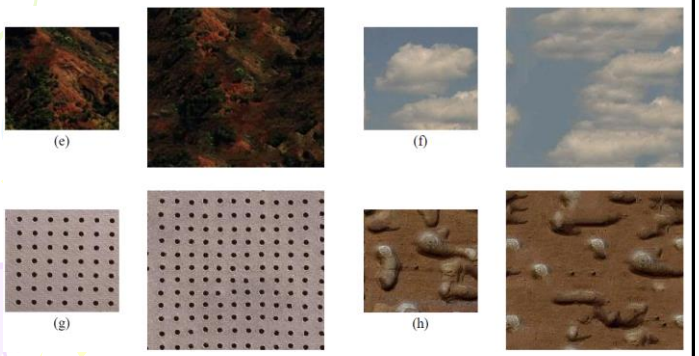
L. Wei, M. Levoy, “Fast Texture Synthesis using Tree-structured Vector Quantization”, 2000

実験結果



L. Wei, M. Levoy, “Fast Texture Synthesis using Tree-structured Vector Quantization”, 2000

実験結果



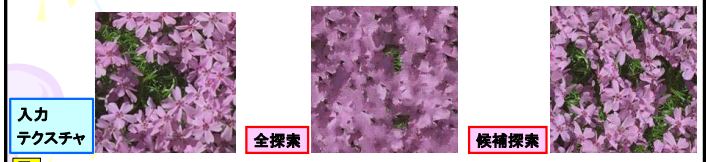
L. Wei, M. Levoy, “Fast Texture Synthesis using Tree-structured Vector Quantization”, 2000

実験結果

L. Wei, M. Levoy, “Fast Texture Synthesis using Tree-structured Vector Quantization”, 2000

候補探索アルゴリズム

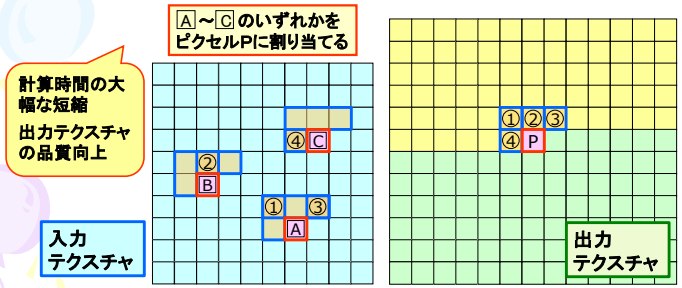
- **L. Wei, M. Levoy (2000)** による基本アルゴリズム（**全探索アルゴリズム**）の問題点
  - 入力テクスチャの**全てのピクセルを探索** → **計算時間大**
  - スムーズなパターンは得意，エッジ等が多いパターンは苦手
- **M. Ashikhmin (2001)** による**候補探索アルゴリズム**
  - 選抜された**候補ピクセルだけを探索** → **計算時間の大幅な短縮**
  - エッジ等が多いパターンに対しても良好な合成結果
- **M. Ashikhmin, “Synthesizing Natural Textures”, 2001**



M. Ashikhmin, “Synthesizing Natural Textures”, 2001

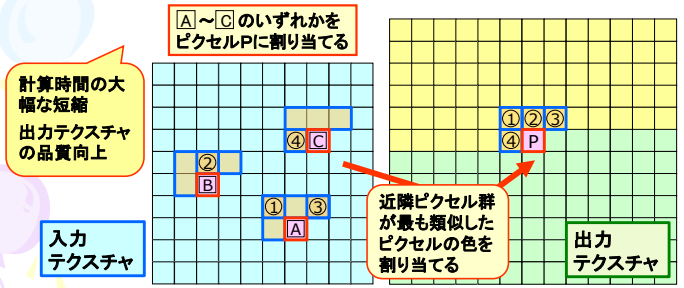
候補探索アルゴリズム

- 入力テクスチャ上の**隣接するピクセル群**をそのまま出力テクスチャに割り当てることで，切れ目のない自然なテクスチャを合成する．
- 出力テクスチャの処理中のピクセルPについて，その近隣ピクセル群中の（合成済みの）各ピクセルに入力テクスチャのどのピクセル（の色）が割り当てられたかという情報を用い，ピクセルPに割り当てる**候補ピクセルを限定**する．最大候補数は近隣ピクセル群のピクセル数となる．



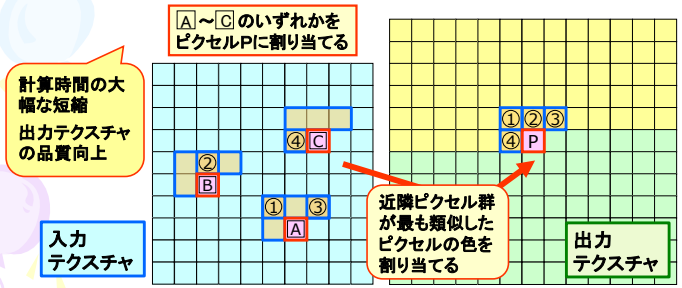
候補探索アルゴリズム

- 候補ピクセルのうち，出力テクスチャのピクセルPと**最も類似した近隣ピクセル群**をもつピクセルを選択し，その色を割り当てる．
- 類似度の評価には**Wei, Levoy**の全探索アルゴリズムと同じ評価関数を用いる．



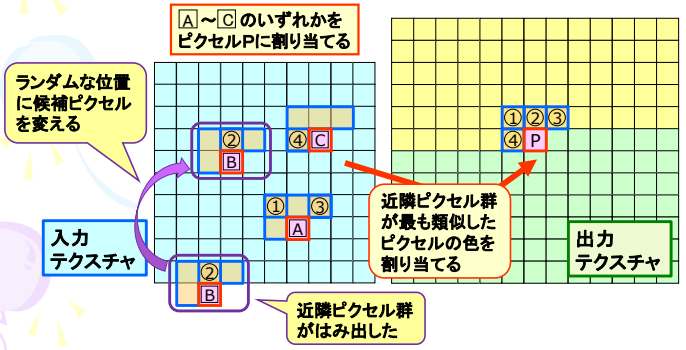
候補探索アルゴリズム：候補の記憶

- 出力テクスチャのサイズの**座標記憶用配列**を用意し，出力テクスチャのピクセルに入力テクスチャのピクセルを割り当てる際，割り当てたピクセル座標を記憶しておく．
- 座標記憶用配列はランダムなピクセル座標で初期化しておく．



候補探索アルゴリズム：近隣ピクセル群

- 近隣ピクセル群が入力テクスチャ内に収まるピクセルだけを候補にする．
- 近隣ピクセル群が入力テクスチャの外にはみ出した場合，入力テクスチャ内に収まるランダムな位置に候補ピクセルを変える．



候補探索アルゴリズム：合成の傾向

- 入力テクスチャ上の**隣接するピクセル群**が**一塊の（不規則な境界を持つ）ピース**として出力テクスチャに割り当てられる．
- 近隣ピクセル群が入力テクスチャの（下端で）外にはみ出した候補ピクセルをランダム位置に変更
  → ピースの下端が平行で，隣接するピースと不連続な色になる傾向大

  入力ピクセルの下端に近い候補ピクセルを確率的にランダム位置に変更．下端に近いほど変更確率を上げる．

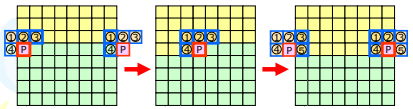
M. Ashikhmin, “Synthesizing Natural Textures”, 2001

候補探索アルゴリズム：多重解像度処理

- **Wei, Levoy**の全探索アルゴリズム
  - 近隣ピクセル群のサイズが大きなほど → 合成結果は良好
    → 計算時間は増加
  - 多重解像度処理により，近隣ピクセル群のサイズが小さな場合でも，単一解像度でサイズが大きな場合に匹敵する良好な合成結果が得られる． → 計算時間の短縮化
- **Ashikhmin**の候補探索アルゴリズム
  - 自然物などのテクスチャでは，良好な合成結果を得る最適な近隣ピクセル群のサイズは特徴的パターンの構造に依存し，必ずしも大きいほど良いわけではない．
    → **Wei, Levoy**が主張する**多重解像度処理の利点が薄れる**．
    → 単一解像度による実装を行っている．

候補探索アルゴリズム：出力テクスチャの環状処理

- 出力テクスチャの**境界**の**環状化（左端と右端，上端と下端の連続化）**
  ← 近隣ピクセル群の構成を工夫して環状化させることで実現
- **左端と右端の環状化**

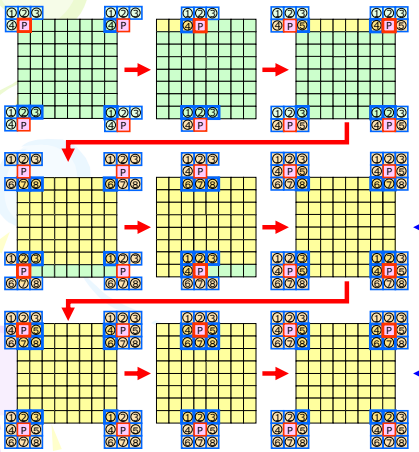


右端の数ピクセル幅（近隣ピクセル群サイズに依存）の合成の際，近隣ピクセル群を環状化させて左端の合成済みピクセルを含める．

- 未処理のピクセル
- 合成済みピクセル
- 処理中のピクセル
- 近隣ピクセル群内のピクセル

候補探索アルゴリズム：出力テクスチャの環状処理

- **上端と下端の環状化**

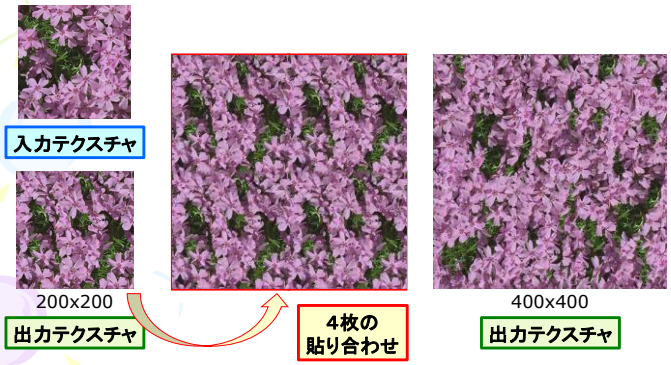


下端の数ピクセル幅（近隣ピクセル群サイズに依存）の合成の際，近隣ピクセル群を環状化させて上端も加え，全ての合成済みピクセルを含める．

最後のピクセルの合成後，再度，上端の数ピクセル幅の合成を行う．その際，近隣ピクセル群を環状化させて下端も加え（正方領域となる），全ての合成済みピクセルを含める．

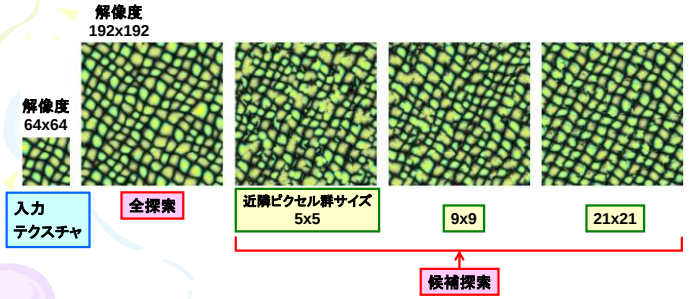
候補探索アルゴリズム：出力テクスチャの環状処理

■ 出力テクスチャの環状処理の結果



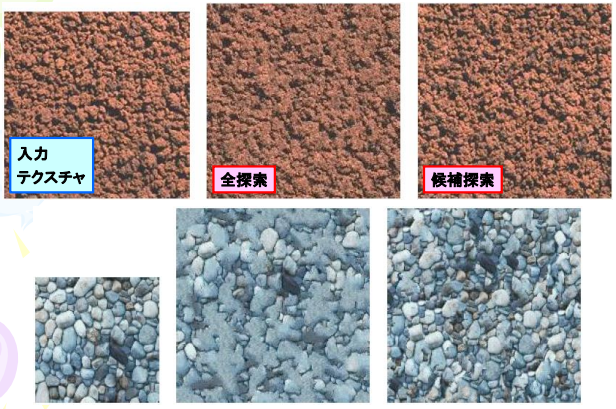
M. Ashikhmin, “Synthesizing Natural Textures”, 2001

候補探索アルゴリズム：実験結果



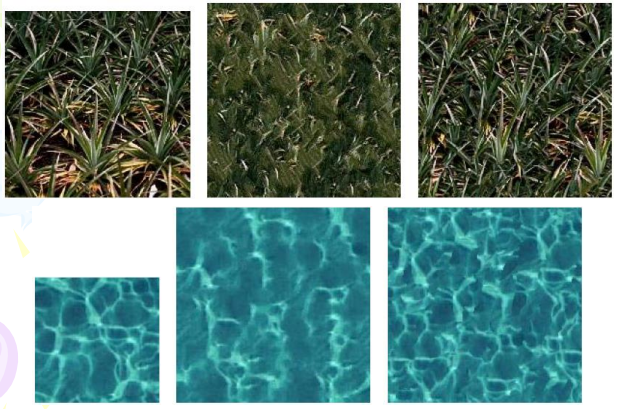
M. Ashikhmin, “Synthesizing Natural Textures”, 2001

候補探索アルゴリズム：実験結果



M. Ashikhmin, “Synthesizing Natural Textures”, 2001

候補探索アルゴリズム：実験結果

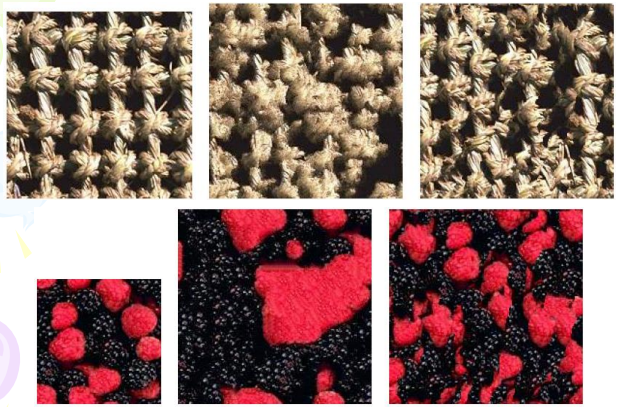


M. Ashikhmin, “Synthesizing Natural Textures”, 2001

候補探索アルゴリズム：実験結果

M. Ashikhmin, “Synthesizing Natural Textures”, 2001

候補探索アルゴリズム：実験結果



M. Ashikhmin, “Synthesizing Natural Textures”, 2001

候補探索アルゴリズム：実験結果

M. Ashikhmin, “Synthesizing Natural Textures”, 2001

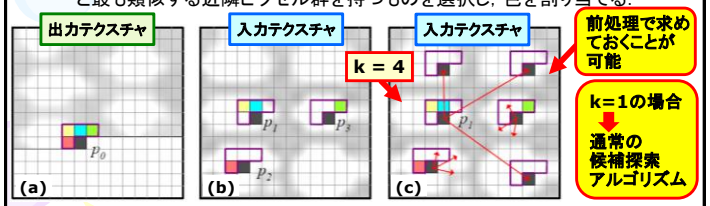
k-コヒーレンス探索アルゴリズム

- X. Tong ら (2002) による k-コヒーレンス探索アルゴリズム
  - M. Ashikhmin (2001) による候補探索アルゴリズムを拡張.
  - 候補ピクセル数を増やし，より高品質な合成を行う.
- X. Tong, J. Zhang, L. Liu, X. Wang, B. Guo, H. Shum, “Synthesis of Bidirectional Texture Functions on Arbitrary Surfaces”, SIGGRAPH 2002

X. Tong, J. Zhang, L. Liu, X. Wang, B. Guo, H. Shum, “Synthesis of Bidirectional Texture Functions on Arbitrary Surfaces”, 2002

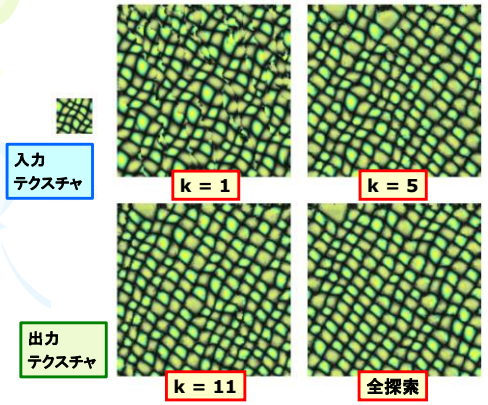
k-コヒーレンス探索アルゴリズム

- 候補探索アルゴリズムと同様の方法で，出力テクスチャの処理中のピクセル$\mathbf{P_0}$について，その近隣ピクセル群により，入力テクスチャ上の候補ピクセル$\mathbf{P_i}$ **(i=1,…,N)**を限定する． → 図**(b)**
- 各候補ピクセル$\mathbf{P_i}$に対して，近隣ピクセル群が類似する**k-1**個のピクセル$\mathbf{P_{i,j}}$ **(j=2,…,k)**を入力テクスチャ上で探索する．そして，候補ピクセル$\mathbf{P_i}$を$\mathbf{P_{i,1}}$とする． → 図**(c)**
- 全ての候補ピクセル$\mathbf{P_{i,j}}$ **(i=1,…,N, j=1,…,k)**のうちで，ピクセル$\mathbf{P_0}$と最も類似する近隣ピクセル群を持つものを選択し，色を割り当てる．



X. Tong, J. Zhang, L. Liu, X. Wang, B. Guo, H. Shum, “Synthesis of Bidirectional Texture Functions on Arbitrary Surfaces”, 2002

k-コヒーレンス探索アルゴリズム：実験結果



X. Tong, J. Zhang, L. Liu, X. Wang, B. Guo, H. Shum, “Synthesis of Bidirectional Texture Functions on Arbitrary Surfaces”, 2002

パッチマッチ探索アルゴリズム

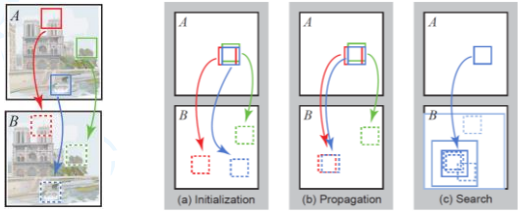
- **テクスチャ合成**等の様々な**イメージエディティング技術**において，**2つの画像間で類似する領域を探索する**ことは重要な基本技術の一つ．
  - テクスチャ合成では，適切な類似領域（近隣ピクセル群）の探索アルゴリズムは画像（テクスチャ）上の特徴的パターンの構造に依存．
    - スムーズなパターン → **全探索**
    - エッジ等が多いパターン → **候補探索**，**k-コヒーレンス探索**
- 特徴的パターン構造によらず，シンプルに類似度評価関数のみによって類似領域を探索する**全探索アルゴリズム**は，基本技術として有用．

  → 全探索アルゴリズムは計算時間大
- **C. Barnes ら (2009)** による**パッチマッチ探索アルゴリズム**
  - 全探索を近似的に高速に行う． ➡ **リアルタイム処理**
- **C. Barnes, E. Shechtman, A. Finkelstein, D. Goldman, “PatchMatch: A Randomized Correspondence Algorithm for Structural Image Editing”, SIGGRAPH 2009**

C. Barnes, E. Shechtman, A. Finkelstein, D. Goldman, “PatchMatch: A Randomized Correspondence Algorithm for Structural Image Editing”, 2009

パッチマッチ探索アルゴリズム

- 画像A，Bについて，画像Aの各ピクセル$\mathbf{P_A}$を中心とした**パッチ（部分領域）**$\mathbf{T_A}$に**（近似的に）最も類似**するパッチ$\mathbf{T_B}$を画像Bから探索する．
- 画像Bのパッチの中心ピクセルを$\mathbf{P_B}$とし，画像A，B上の$\mathbf{P_A}$，$\mathbf{P_B}$のピクセル座標$\mathbf{X_A=(x_A, y_A)}$，$\mathbf{X_B=(x_B, y_B)}$の**オフセット** $\mathbf{f=X_B-X_A}$ をピクセル$\mathbf{P_A}$に持たせる．
- パッチマッチ探索アルゴリズムでは，各ピクセル$\mathbf{P_A}$の**オフセットを反復処理で更新**していくことで，最類似パッチの全探索を近似的に高速に行う．

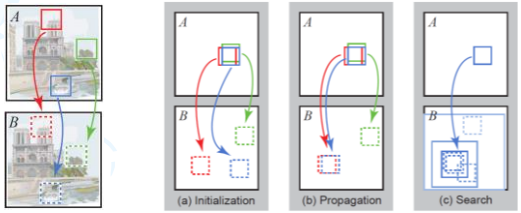


C. Barnes, E. Shechtman, A. Finkelstein, D. Goldman, “PatchMatch: A Randomized Correspondence Algorithm for Structural Image Editing”, 2009

パッチマッチ探索アルゴリズム

- パッチマッチ探索アルゴリズムの構成
  - **初期化(initialization)**
  - 反復処理(収束するまで)
    - 画像Aのピクセル$\mathbf{P_A}$ごとの処理
      - **伝播(propagation)**
      - **ランダム探索(random search)**

通常，反復ごとに
走査順序を逆にする.

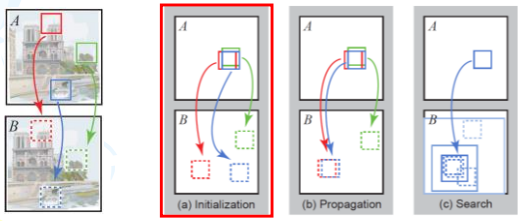


C. Barnes, E. Shechtman, A. Finkelstein, D. Goldman, "PatchMatch: A Randomized Correspondence Algorithm for Structural Image Editing", 2009

パッチマッチ探索アルゴリズム

- パッチマッチ探索アルゴリズムの構成
  - **初期化(initialization)**
  - 反復処理(収束するまで)
    - 画像Aのピクセル$\mathbf{P_A}$ごとの処理
      - **伝播(propagation)**
      - **ランダム探索(random search)**

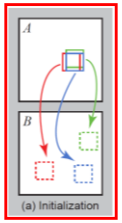
通常，反復ごとに
走査順序を逆にする.



C. Barnes, E. Shechtman, A. Finkelstein, D. Goldman, "PatchMatch: A Randomized Correspondence Algorithm for Structural Image Editing", 2009

パッチマッチ探索アルゴリズム

- **初期化(initialization)**
  - 画像Aの各ピクセル$\mathbf{P_A}$のオフセット**f**をランダムに初期化する.

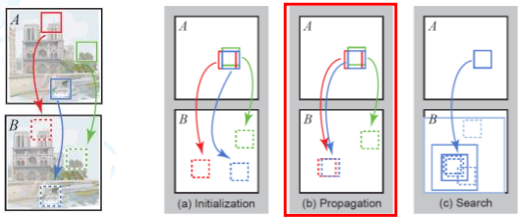


C. Barnes, E. Shechtman, A. Finkelstein, D. Goldman, "PatchMatch: A Randomized Correspondence Algorithm for Structural Image Editing", 2009

パッチマッチ探索アルゴリズム

- パッチマッチ探索アルゴリズムの構成
  - **初期化(initialization)**
  - 反復処理(収束するまで)
    - 画像Aのピクセル$\mathbf{P_A}$ごとの処理
      - **伝播(propagation)**
      - **ランダム探索(random search)**

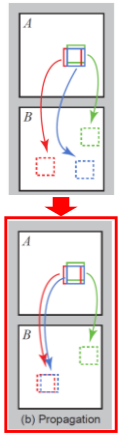
通常，反復ごとに
走査順序を逆にする.



C. Barnes, E. Shechtman, A. Finkelstein, D. Goldman, "PatchMatch: A Randomized Correspondence Algorithm for Structural Image Editing", 2009

パッチマッチ探索アルゴリズム

- **伝播(propagation)**
  - 画像A上でこれから処理する座標$\mathbf{X_A}=(x_A, y_A)$のピクセル$\mathbf{P_A}(\mathbf{X_A})=\mathbf{P_A}(x_A, y_A)$の現在のオフセットを$\mathbf{f}(\mathbf{X_A})=\mathbf{f}(x_A, y_A)$とする.
  - 走査線順で処理済みのピクセル$\mathbf{P_A}(x_A-1, y_A)$, $\mathbf{P_A}(x_A, y_A-1)$の現在のオフセットを$\mathbf{f}(x_A-1, y_A)$, $\mathbf{f}(x_A, y_A-1)$とする.
  - 上記の3つのオフセットのそれぞれについて，座標$\mathbf{X_A}=(x_A, y_A)$に加算して画像B上の座標$\mathbf{X_B}$を求め，画像Bのピクセル$\mathbf{P_B}(\mathbf{X_B})$を中心としたパッチ$\mathbf{T_B}$と画像Aのピクセル$\mathbf{P_A}(x_A, y_A)$を中心としたパッチ$\mathbf{T_A}$の類似度を評価する.
  - 3つのオフセットのうち，最も類似度の高いものによってピクセル$\mathbf{P_A}(x_A, y_A)$のオフセット$\mathbf{f}(x_A, y_A)$を更新する.

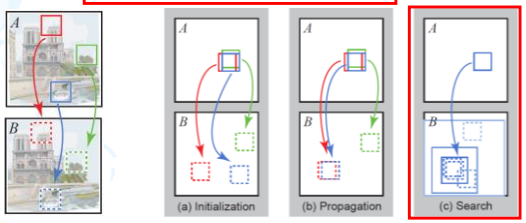


C. Barnes, E. Shechtman, A. Finkelstein, D. Goldman, "PatchMatch: A Randomized Correspondence Algorithm for Structural Image Editing", 2009

パッチマッチ探索アルゴリズム

- パッチマッチ探索アルゴリズムの構成
  - **初期化(initialization)**
  - 反復処理(収束するまで)
    - 画像Aのピクセル$\mathbf{P_A}$ごとの処理
      - **伝播(propagation)**
      - **ランダム探索(random search)**

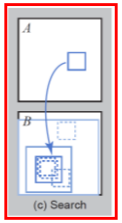
通常，反復ごとに走査順序を逆にする.



C. Barnes, E. Shechtman, A. Finkelstein, D. Goldman, "PatchMatch: A Randomized Correspondence Algorithm for Structural Image Editing", 2009

パッチマッチ探索アルゴリズム

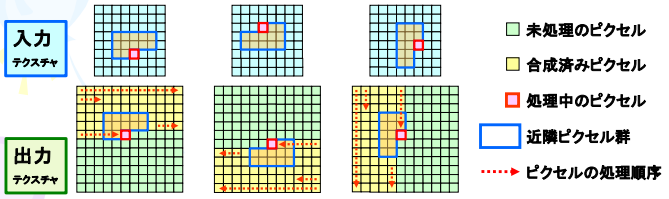
- **ランダム探索(random search)**
  - 画像A上の処理中のピクセル$\mathbf{P_A}(\mathbf{X_A})$の現在のオフセットを$\mathbf{f}(\mathbf{X_A})$とする.
  - そのオフセットによる画像B上のピクセル$\mathbf{P_B}(\mathbf{X_B})$を中心として，周囲の幾つかのピクセルをランダムに選び，各ピクセル(ピクセル$\mathbf{P_B}(\mathbf{X_B})$も含めて)を中心としたパッチ$\mathbf{T_B}$と画像Aのピクセル$\mathbf{P_A}(\mathbf{X_A})$を中心としたパッチ$\mathbf{T_A}$の類似度を評価する.
  - 上記の画像B上の全てのピクセルのうち，最も類似度の高いものによってピクセル$\mathbf{P_A}(\mathbf{X_A})$のオフセット$\mathbf{f}(\mathbf{X_A})$を更新する.



C. Barnes, E. Shechtman, A. Finkelstein, D. Goldman, "PatchMatch: A Randomized Correspondence Algorithm for Structural Image Editing", 2009

ピクセル処理順に非依存のテクスチャ合成

- 近隣ピクセル群の類似度評価によるピクセルベースのテクスチャ合成
  - 近隣ピクセル群は合成済みピクセルだけから構成
  - **近隣ピクセル群が出力テクスチャ上のピクセルの処理順序に依存**し，処理中のピクセルを中心とした非対称(L型)な形状となる.
  - **合成結果がピクセルの処理順序に依存して変わる.**

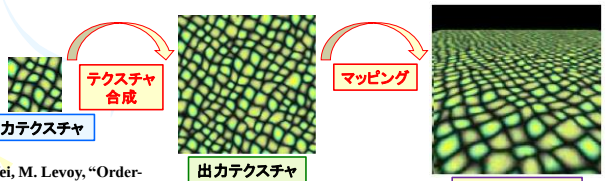


L. Wei, M. Levoy, "Order-Independent Texture Synthesis", 2002

ピクセル処理順に非依存のテクスチャ合成

- 近隣ピクセル群の類似度評価によるピクセルベースのテクスチャ合成
  - ➡ **合成結果がピクセルの処理順序に依存して変わる.**
- レンダリングにおける出力テクスチャの実際の使用
  - 3次元の物体モデル(ポリゴン, 曲面)上にマッピング
  - ラスタ変換(レイトレーシング, Zバッファなど)により, **画像上のピクセル**ごとの物体表面のテクスチャ色を求める.
  - **出力テクスチャ上のピクセル**のうち, **どれがどの順序で必要になるか不定**
- ➡ **合成のたびにテクスチャが変わってしまう.**

意味の違い！
テクスチャのピクセル
画像のピクセル



L. Wei, M. Levoy, "Order-Independent Texture Synthesis", 2002

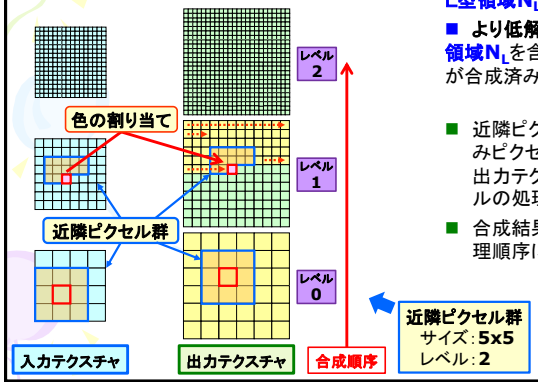
ピクセル処理順に非依存のテクスチャ合成

- **案1**: 出力テクスチャ全体(全ピクセルの色)をあらかじめ合成しておく.
  - 全ピクセルの色が必要なわけではない.
  - 視点と物体の距離によるLODのための**多重解像度化(ミップマップ)**
    → 使わないピクセルの合成は非常に無駄！
  - ➡ 計算時間大
- **案2**: 必要なピクセルの色だけを合成する.
  - 計算時間の短縮化の可能性あり
  - **合成結果が(必要に応じて合成する)ピクセルの処理順序に依存して変わる**という問題あり.
- **L. Wei, M. Levoy (2002)** による**ピクセル処理順に非依存のテクスチャ合成 → 多重解像度処理を複数パス(世代)**で実行
- **L. Wei, M. Levoy, "Order-Independent Texture Synthesis", Stanford Computer Science TR-2002-01**

L. Wei, M. Levoy, "Order-Independent Texture Synthesis", 2002

ピクセル処理順に非依存のテクスチャ合成

- **通常の多重解像度処理**
  近隣ピクセル群を複数の**解像度レベル**で構成.
- **処理中のレベル**:
  合成済みピクセルだけを含む**L型領域N_L**
- **より低解像度のレベル**:
  領域N_Lを含む**正方領域**(全てが合成済みピクセル)
- 近隣ピクセル群は合成済みピクセルだけで構成され, 出力テクスチャ上のピクセルの処理順序に依存.
- 合成結果がピクセルの処理順序に依存して変わる.



ピクセル処理順に非依存のテクスチャ合成

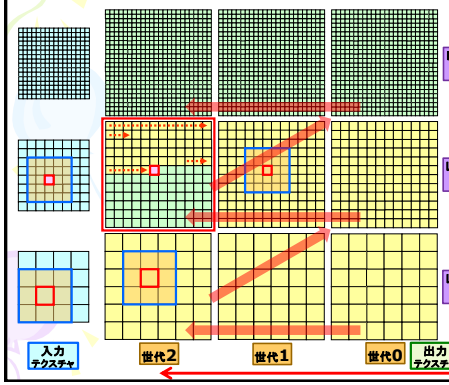
- **ピクセル処理順に非依存のテクスチャ合成**
  解像度レベルと**世代**による複数の出力テクスチャ

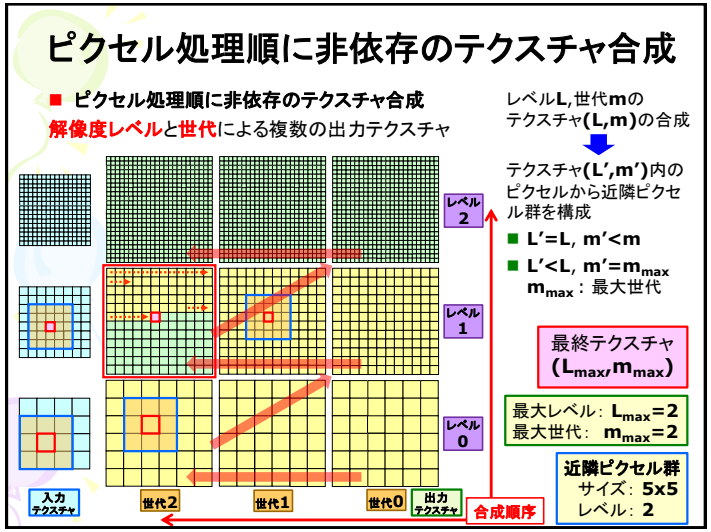
解像度レベル・世代方向に別の出力テクスチャを**段階的に切り替えて**合成を進める.

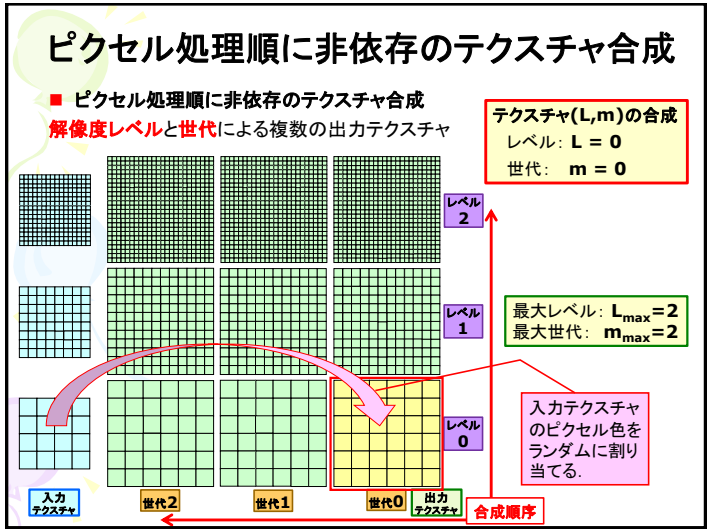
低レベル・世代から高レベル・世代に向けて合成.

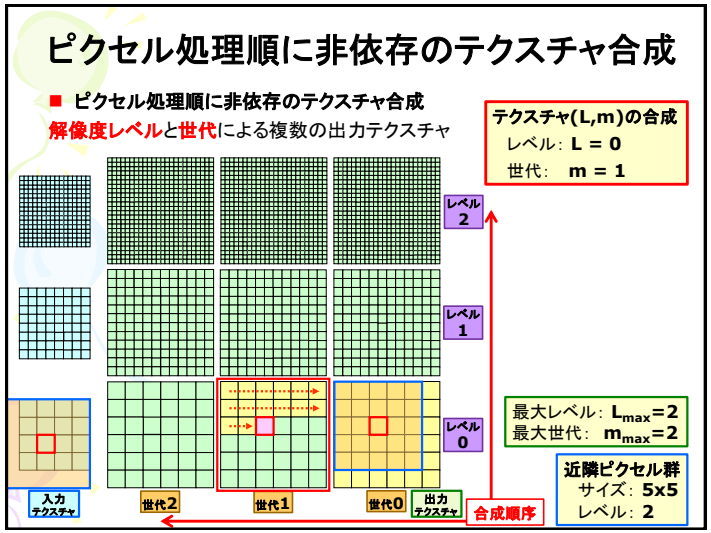
- **合成済みピクセル色を保持するテクスチャ領域 → 正方型の近隣ピクセル群**を構成して類似度評価に使う.
- **その段階で合成するピクセル色を格納するテクスチャ領域**

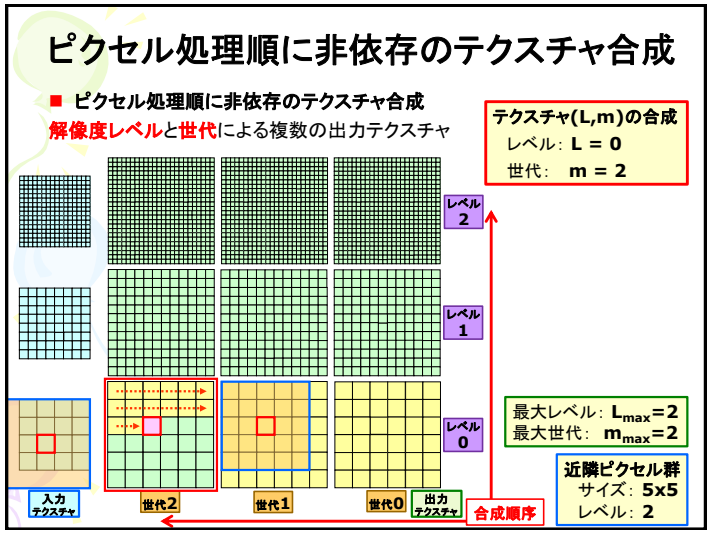
近隣ピクセル群が**ピクセルの処理順序に非依存**
→ **合成結果**が**ピクセルの処理順序に非依存**

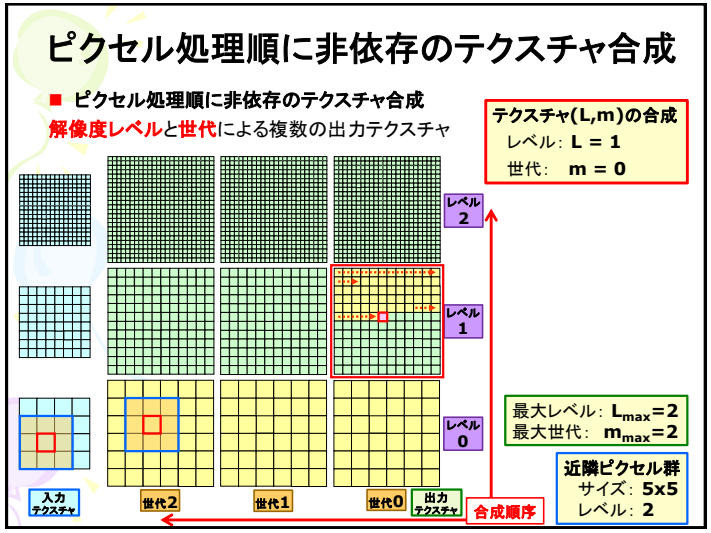


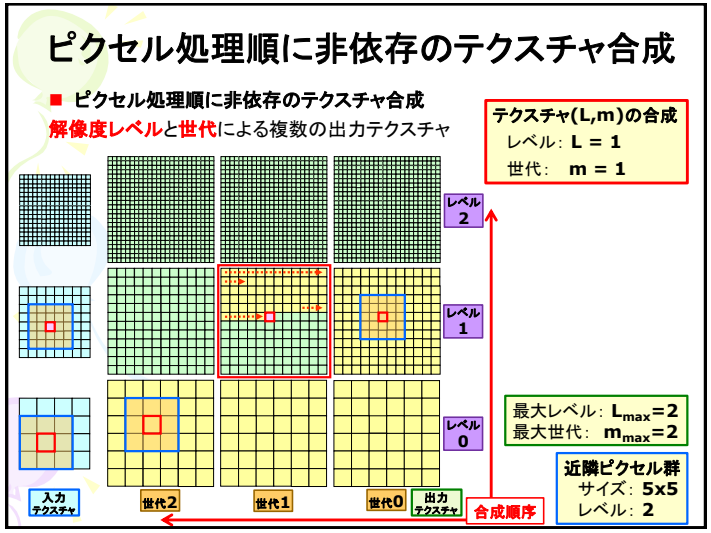

ピクセル処理順に非依存のテクスチャ合成
■ ピクセル処理順に非依存のテクスチャ合成
解像度レベルと世代による複数の出力テクスチャ
レベルL,世代mのテクスチャ(L,m)の合成
テクスチャ(L',m')内のピクセルから近隣ピクセル群を構成
■ L'=L, m'<m
■ L'<L, m'=mmax
mmax：最大世代
最終テクスチャ (Lmax,mmax)
最大レベル：Lmax=2
最大世代：mmax=2
近隣ピクセル群
サイズ：5x5
レベル：2
レベル2
レベル1
レベル0
入力テクスチャ
世代2
世代1
世代0
出力テクスチャ
合成順序

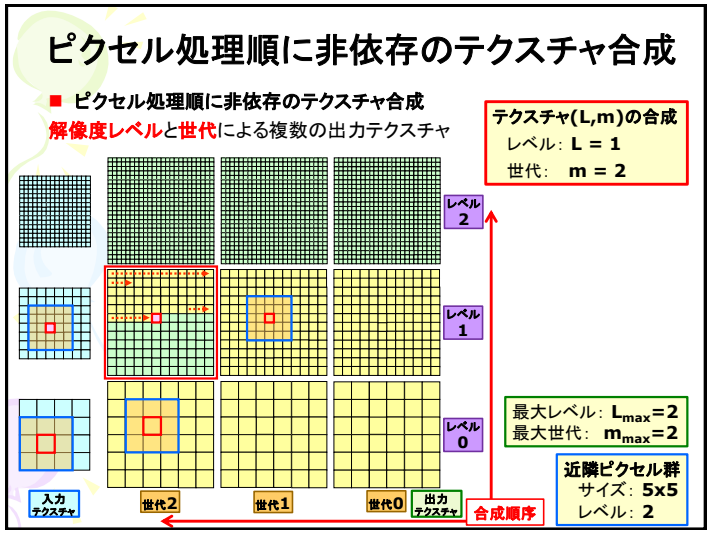

ピクセル処理順に非依存のテクスチャ合成
■ ピクセル処理順に非依存のテクスチャ合成
解像度レベルと世代による複数の出力テクスチャ
テクスチャ(L,m)の合成
レベル：L = 0
世代：m = 0
最大レベル：Lmax=2
最大世代：mmax=2
入力テクスチャのピクセル色をランダムに割り当てる.
レベル2
レベル1
レベル0
入力テクスチャ
世代2
世代1
世代0
出力テクスチャ
合成順序

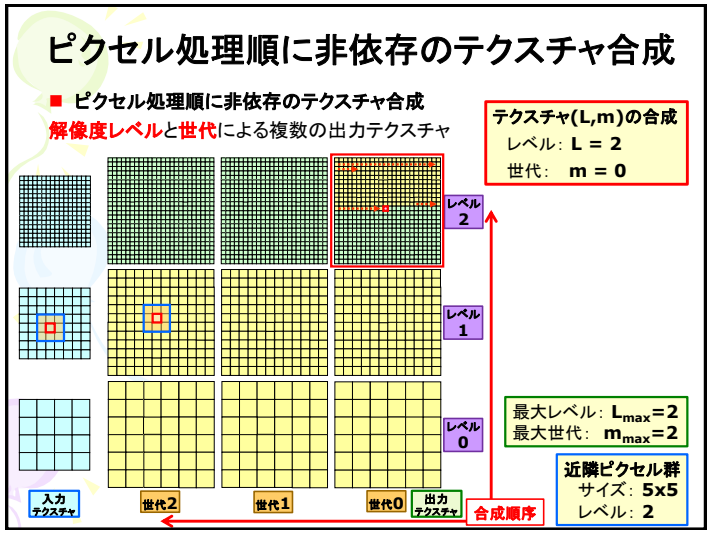

ピクセル処理順に非依存のテクスチャ合成
■ ピクセル処理順に非依存のテクスチャ合成
解像度レベルと世代による複数の出力テクスチャ
テクスチャ(L,m)の合成
レベル：L = 0
世代：m = 1
最大レベル：Lmax=2
最大世代：mmax=2
近隣ピクセル群
サイズ：5x5
レベル：2
レベル2
レベル1
レベル0
入力テクスチャ
世代2
世代1
世代0
出力テクスチャ
合成順序


ピクセル処理順に非依存のテクスチャ合成
■ ピクセル処理順に非依存のテクスチャ合成
解像度レベルと世代による複数の出力テクスチャ
テクスチャ(L,m)の合成
レベル：L = 0
世代：m = 2
最大レベル：Lmax=2
最大世代：mmax=2
近隣ピクセル群
サイズ：5x5
レベル：2
レベル2
レベル1
レベル0
入力テクスチャ
世代2
世代1
世代0
出力テクスチャ
合成順序


ピクセル処理順に非依存のテクスチャ合成
■ ピクセル処理順に非依存のテクスチャ合成
解像度レベルと世代による複数の出力テクスチャ
テクスチャ(L,m)の合成
レベル: L = 1
世代: m = 0
レベル 2
レベル 1
レベル 0
最大レベル: Lmax=2
最大世代: mmax=2
近隣ピクセル群
サイズ: 5x5
レベル: 2
入力テクスチャ
世代2
世代1
世代0
出力テクスチャ
合成順序


ピクセル処理順に非依存のテクスチャ合成
■ ピクセル処理順に非依存のテクスチャ合成
解像度レベルと世代による複数の出力テクスチャ
テクスチャ(L,m)の合成
レベル: L = 1
世代: m = 1
レベル 2
レベル 1
レベル 0
最大レベル: Lmax=2
最大世代: mmax=2
近隣ピクセル群
サイズ: 5x5
レベル: 2
入力テクスチャ
世代2
世代1
世代0
出力テクスチャ
合成順序

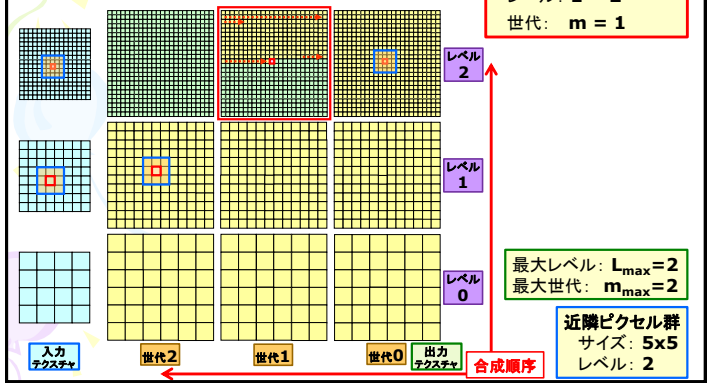

ピクセル処理順に非依存のテクスチャ合成
■ ピクセル処理順に非依存のテクスチャ合成
解像度レベルと世代による複数の出力テクスチャ
テクスチャ(L,m)の合成
レベル: L = 1
世代: m = 2
レベル 2
レベル 1
レベル 0
最大レベル: Lmax=2
最大世代: mmax=2
近隣ピクセル群
サイズ: 5x5
レベル: 2
入力テクスチャ
世代2
世代1
世代0
出力テクスチャ
合成順序


ピクセル処理順に非依存のテクスチャ合成
■ ピクセル処理順に非依存のテクスチャ合成
解像度レベルと世代による複数の出力テクスチャ
テクスチャ(L,m)の合成
レベル: L = 2
世代: m = 0
レベル 2
レベル 1
レベル 0
最大レベル: Lmax=2
最大世代: mmax=2
近隣ピクセル群
サイズ: 5x5
レベル: 2
入力テクスチャ
世代2
世代1
世代0
出力テクスチャ
合成順序

ピクセル処理順に非依存のテクスチャ合成

■ ピクセル処理順に非依存のテクスチャ合成

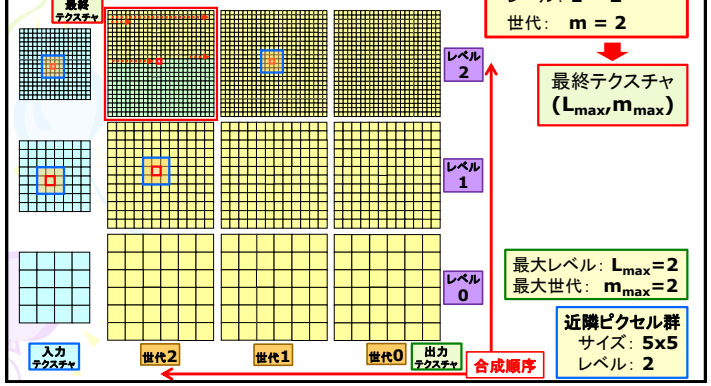
解像度レベルと世代による複数の出力テクスチャ



ピクセル処理順に非依存のテクスチャ合成

■ ピクセル処理順に非依存のテクスチャ合成

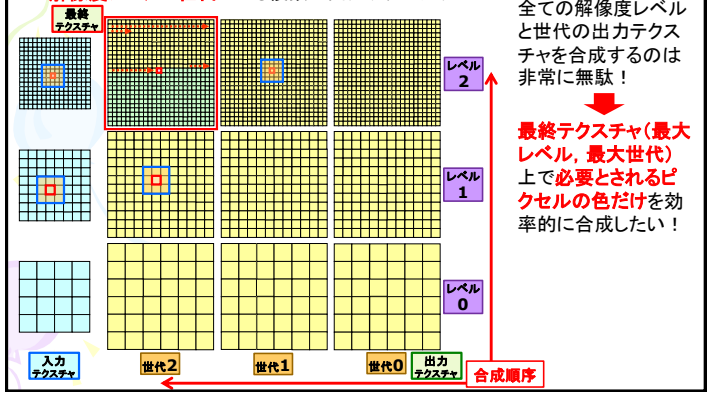
解像度レベルと世代による複数の出力テクスチャ



ピクセル処理順に非依存のテクスチャ合成

■ ピクセル処理順に非依存のテクスチャ合成

解像度レベルと世代による複数の出力テクスチャ



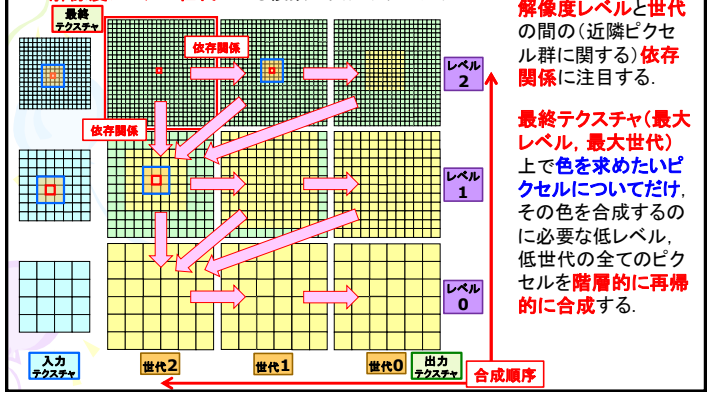
全ての解像度レベルと世代の出力テクスチャを合成するのは非常に無駄！

最終テクスチャ(最大レベル，最大世代)上で必要とされるピクセルの色だけを効率的に合成したい！

ピクセル処理順に非依存のテクスチャ合成

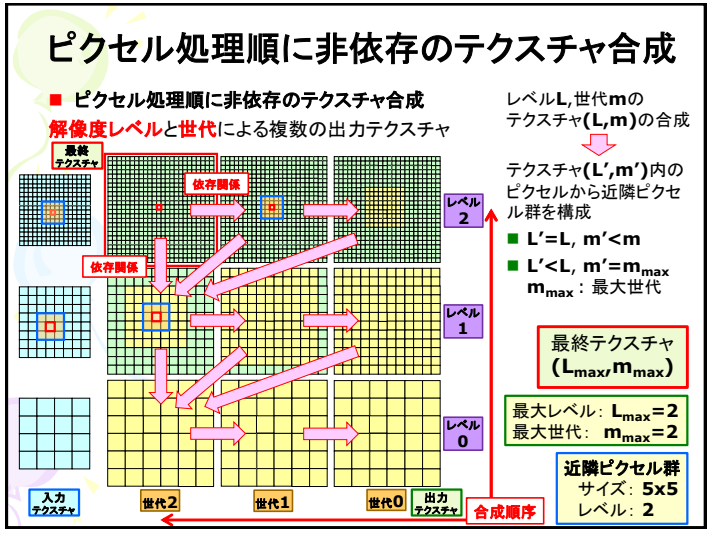
■ ピクセル処理順に非依存のテクスチャ合成

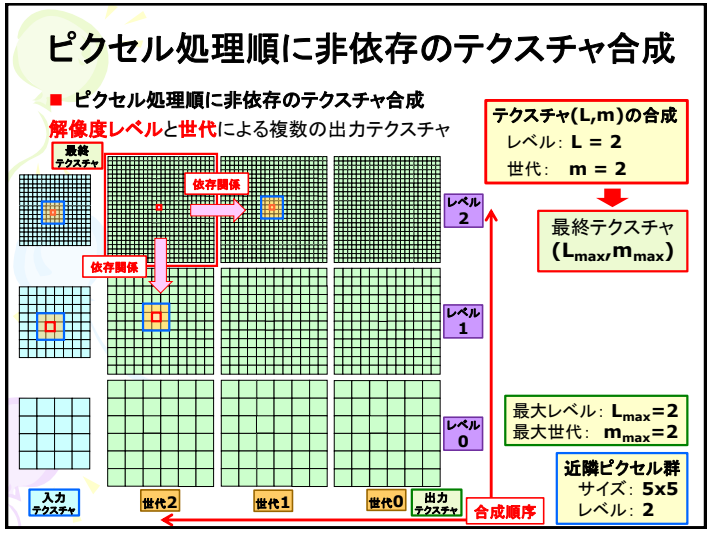
解像度レベルと世代による複数の出力テクスチャ

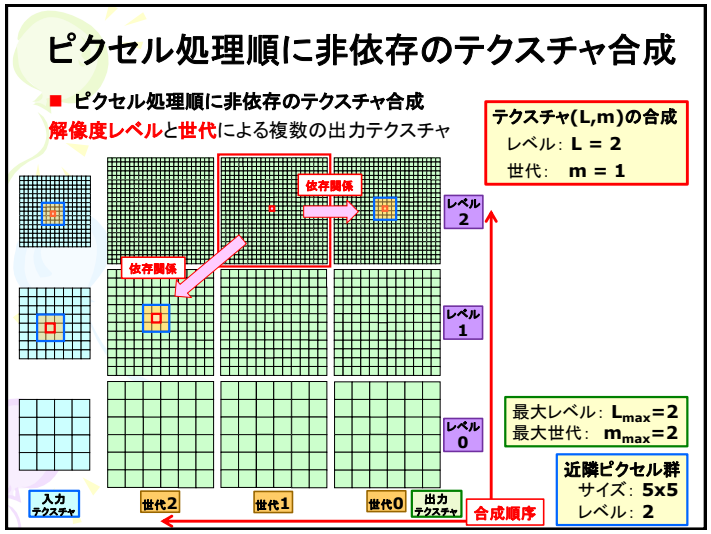


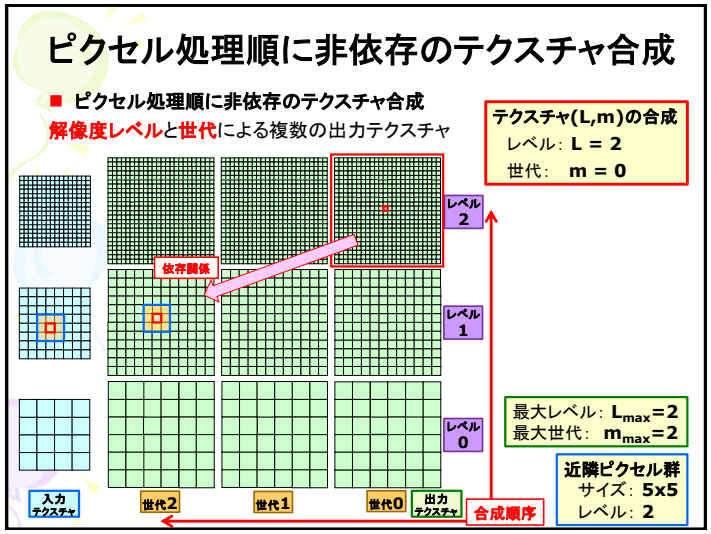
解像度レベルと**世代**の間の(近隣ピクセル群に関する)**依存関係**に注目する.

最終テクスチャ(最大レベル，最大世代)上で色を求めたいピクセルについてだけ，その色を合成するのに必要な低レベル，低世代の全てのピクセルを**階層的に再帰的に合成**する.

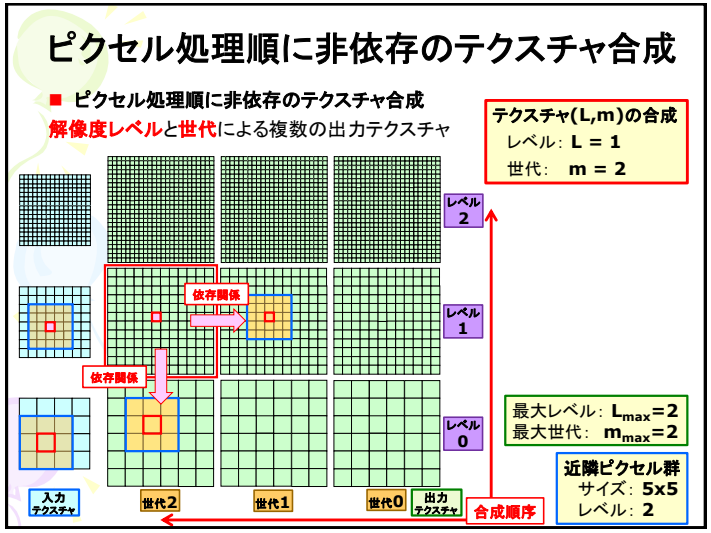


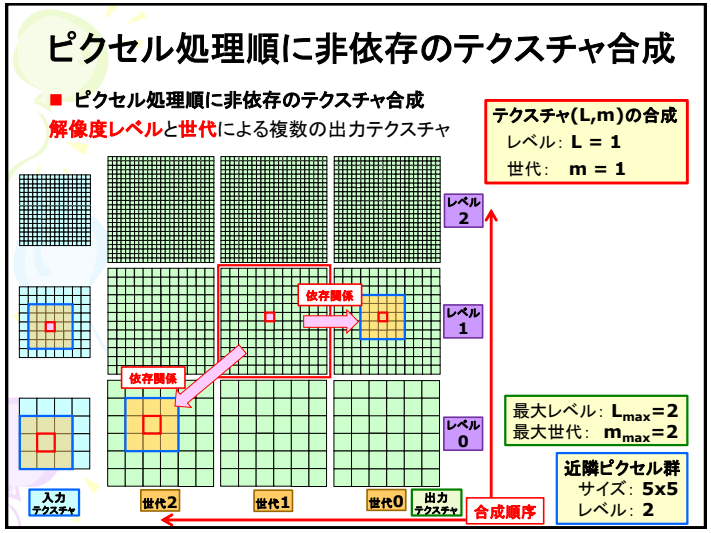


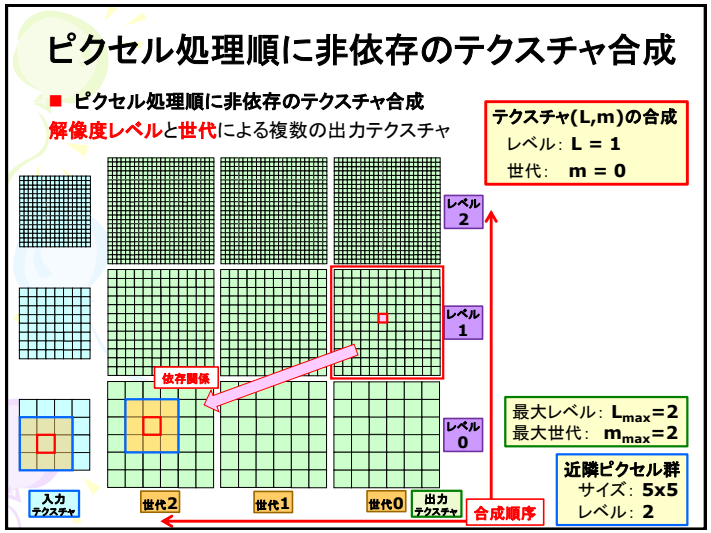


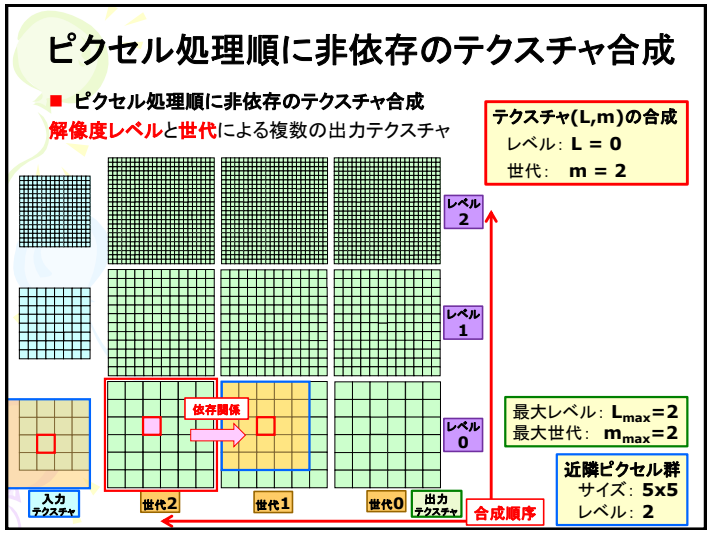


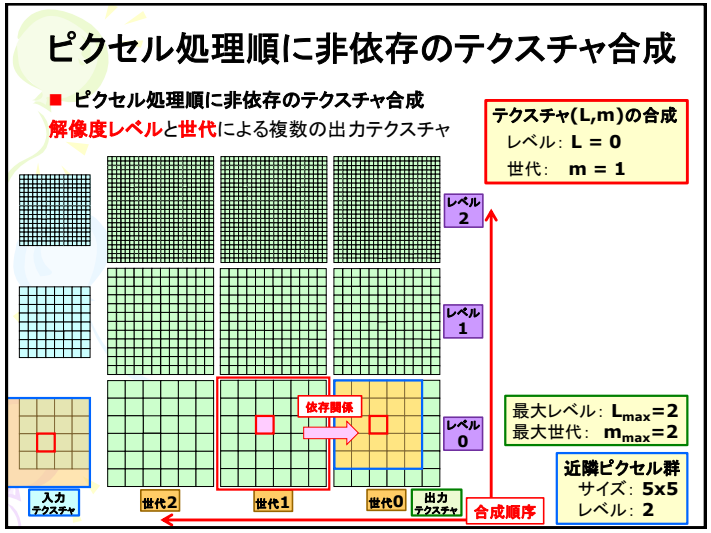
イメージシンセシス　テクスチャ合成(ピクセルベース手法)
藤本忠博

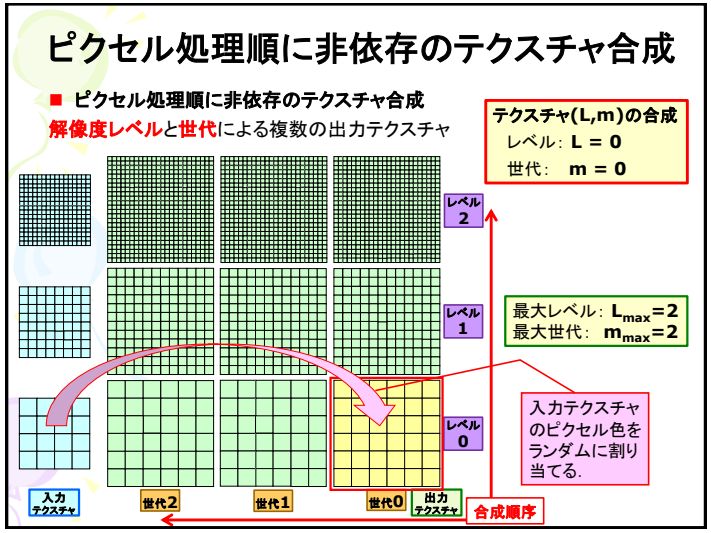

ピクセル処理順に非依存のテクスチャ合成
■ ピクセル処理順に非依存のテクスチャ合成
解像度レベルと世代による複数の出力テクスチャ
テクスチャ(L,m)の合成
レベル: L = 1
世代: m = 2
レベル 2
レベル 1
レベル 0
依存関係
依存関係
最大レベル: Lmax=2
最大世代: mmax=2
近隣ピクセル群
サイズ: 5x5
レベル: 2
入力テクスチャ
世代2
世代1
世代0
出力テクスチャ
合成順序

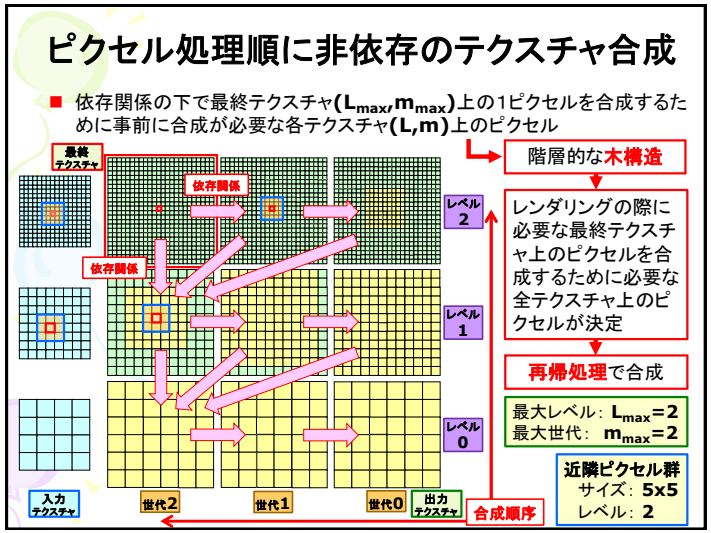

ピクセル処理順に非依存のテクスチャ合成
■ ピクセル処理順に非依存のテクスチャ合成
解像度レベルと世代による複数の出力テクスチャ
テクスチャ(L,m)の合成
レベル: L = 1
世代: m = 1
レベル 2
レベル 1
レベル 0
依存関係
依存関係
最大レベル: Lmax=2
最大世代: mmax=2
近隣ピクセル群
サイズ: 5x5
レベル: 2
入力テクスチャ
世代2
世代1
世代0
出力テクスチャ
合成順序

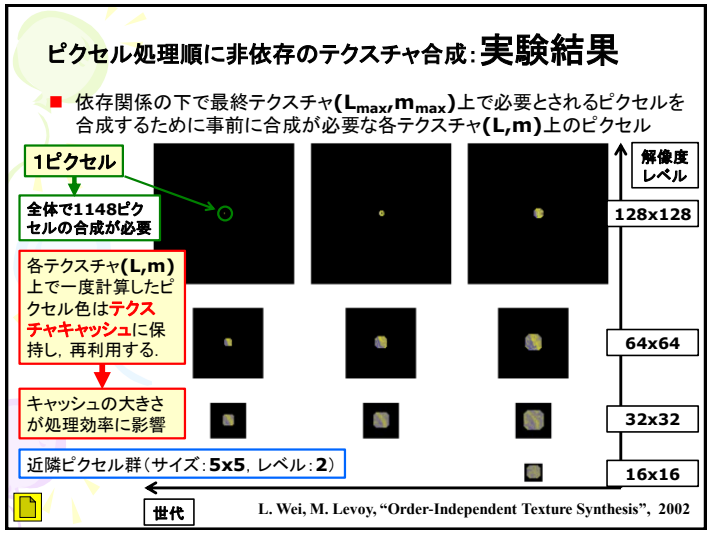

ピクセル処理順に非依存のテクスチャ合成
■ ピクセル処理順に非依存のテクスチャ合成
解像度レベルと世代による複数の出力テクスチャ
テクスチャ(L,m)の合成
レベル: L = 1
世代: m = 0
レベル 2
レベル 1
レベル 0
依存関係
最大レベル: Lmax=2
最大世代: mmax=2
近隣ピクセル群
サイズ: 5x5
レベル: 2
入力テクスチャ
世代2
世代1
世代0
出力テクスチャ
合成順序

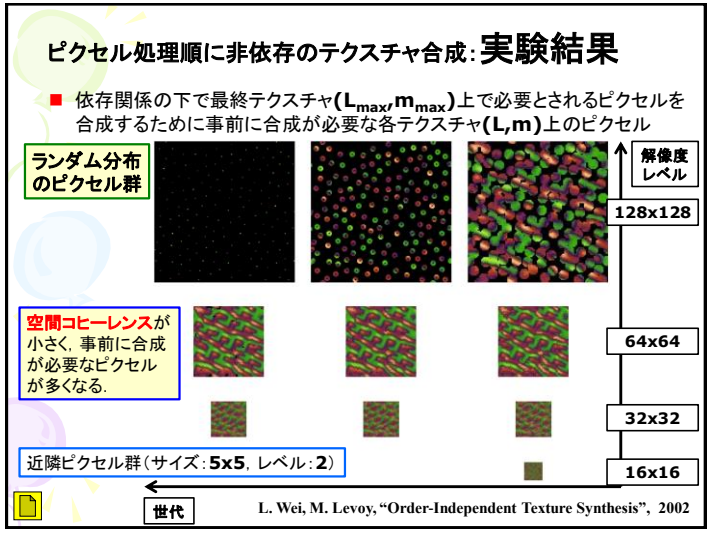

ピクセル処理順に非依存のテクスチャ合成
■ ピクセル処理順に非依存のテクスチャ合成
解像度レベルと世代による複数の出力テクスチャ
テクスチャ(L,m)の合成
レベル: L = 0
世代: m = 2
レベル 2
レベル 1
レベル 0
依存関係
最大レベル: Lmax=2
最大世代: mmax=2
近隣ピクセル群
サイズ: 5x5
レベル: 2
入力テクスチャ
世代2
世代1
世代0
出力テクスチャ
合成順序

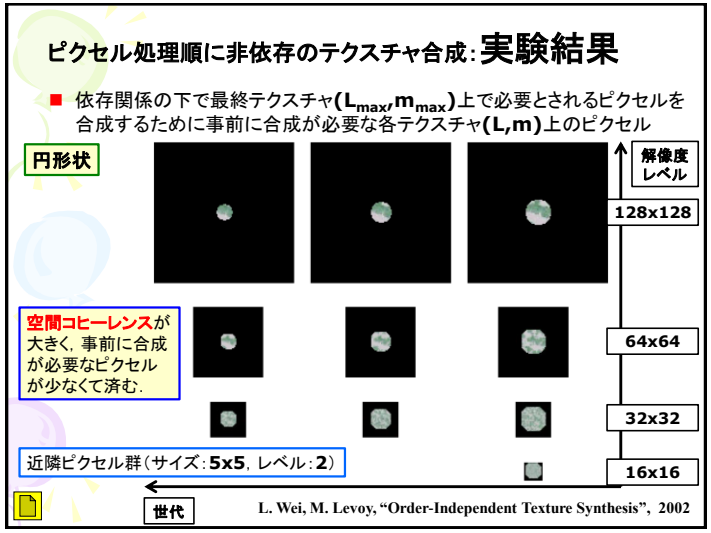

ピクセル処理順に非依存のテクスチャ合成
■ ピクセル処理順に非依存のテクスチャ合成
解像度レベルと世代による複数の出力テクスチャ
テクスチャ(L,m)の合成
レベル: L = 0
世代: m = 1
レベル 2
レベル 1
レベル 0
依存関係
最大レベル: Lmax=2
最大世代: mmax=2
近隣ピクセル群
サイズ: 5x5
レベル: 2
入力テクスチャ
世代2
世代1
世代0
出力テクスチャ
合成順序

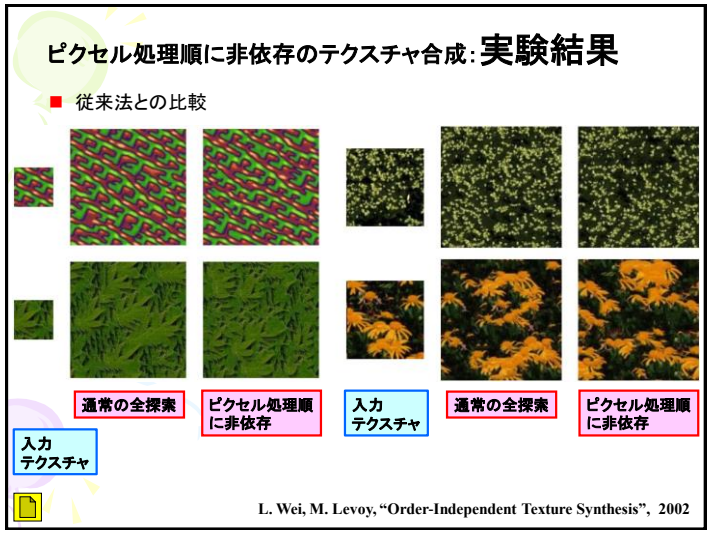

ピクセル処理順に非依存のテクスチャ合成
■ ピクセル処理順に非依存のテクスチャ合成
解像度レベルと世代による複数の出力テクスチャ
テクスチャ(L,m)の合成
レベル: L = 0
世代: m = 0
レベル 2
レベル 1
レベル 0
最大レベル: Lmax=2
最大世代: mmax=2
入力テクスチャのピクセル色をランダムに割り当てる.
入力テクスチャ
世代2
世代1
世代0
出力テクスチャ
合成順序

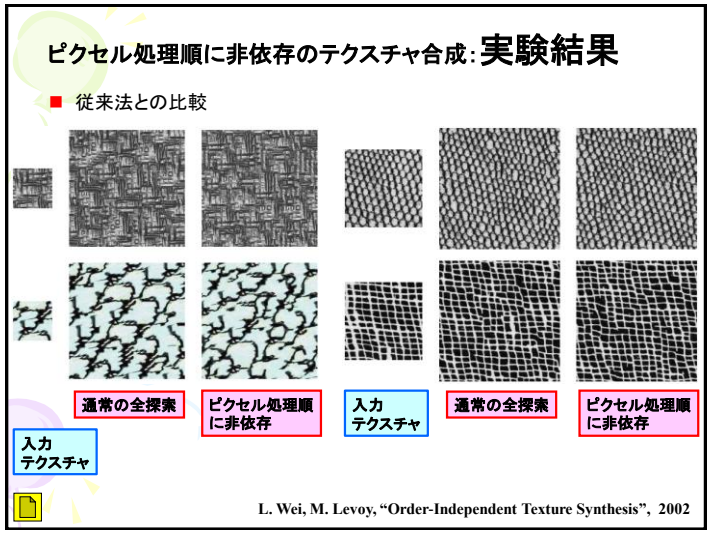

ピクセル処理順に非依存のテクスチャ合成
■ 依存関係の下で最終テクスチャ(Lmax,mmax)上の1ピクセルを合成するために事前に合成が必要な各テクスチャ(L,m)上のピクセル
最終テクスチャ
依存関係
依存関係
階層的な木構造
レンダリングの際に必要な最終テクスチャ上のピクセルを合成するために必要な全テクスチャ上のピクセルが決定
再帰処理で合成
レベル 2
レベル 1
レベル 0
最大レベル: Lmax=2
最大世代: mmax=2
近隣ピクセル群
サイズ: 5x5
レベル: 2
入力テクスチャ
世代2
世代1
世代0
出力テクスチャ
合成順序


ピクセル処理順に非依存のテクスチャ合成:実験結果
■ 依存関係の下で最終テクスチャ(Lmax,mmax)上で必要とされるピクセルを合成するために事前に合成が必要な各テクスチャ(L,m)上のピクセル
1ピクセル
全体で1148ピクセルの合成が必要
各テクスチャ(L,m)上で一度計算したピクセル色はテクスチャキャッシュに保持し,再利用する.
キャッシュの大きさが処理効率に影響
近隣ピクセル群(サイズ:5x5, レベル:2)
解像度レベル
128x128
64x64
32x32
16x16
世代
L. Wei, M. Levoy, "Order-Independent Texture Synthesis", 2002


ピクセル処理順に非依存のテクスチャ合成：実験結果
依存関係の下で最終テクスチャ(L_{max},m_{max})上で必要とされるピクセルを合成するために事前に合成が必要な各テクスチャ(L,m)上のピクセル
ランダム分布のピクセル群
解像度レベル
128x128
64x64
32x32
16x16
空間コヒーレンスが小さく，事前に合成が必要なピクセルが多くなる．
近隣ピクセル群（サイズ：5x5，レベル：2）
世代
L. Wei, M. Levoy, "Order-Independent Texture Synthesis", 2002


ピクセル処理順に非依存のテクスチャ合成：実験結果
依存関係の下で最終テクスチャ(L_{max},m_{max})上で必要とされるピクセルを合成するために事前に合成が必要な各テクスチャ(L,m)上のピクセル
円形状
解像度レベル
128x128
64x64
32x32
16x16
空間コヒーレンスが大きく，事前に合成が必要なピクセルが少なくて済む．
近隣ピクセル群（サイズ：5x5，レベル：2）
世代
L. Wei, M. Levoy, "Order-Independent Texture Synthesis", 2002


ピクセル処理順に非依存のテクスチャ合成：実験結果
従来法との比較
通常の全探索
ピクセル処理順に非依存
入力テクスチャ
通常の全探索
ピクセル処理順に非依存
入力テクスチャ
L. Wei, M. Levoy, "Order-Independent Texture Synthesis", 2002


ピクセル処理順に非依存のテクスチャ合成：実験結果
従来法との比較
通常の全探索
ピクセル処理順に非依存
入力テクスチャ
通常の全探索
ピクセル処理順に非依存
入力テクスチャ
L. Wei, M. Levoy, "Order-Independent Texture Synthesis", 2002

ピクセル処理順に非依存のテクスチャ合成：実験結果

- LODによるミップマップの利用

テクスチャマッピングによるレンダリング画像

ミップマップ（4レベル）
灰色：各レベルでレンダリングに必要とされたピクセル

L. Wei, M. Levoy, "Order-Independent Texture Synthesis", 2002

ユーザ制御アルゴリズム

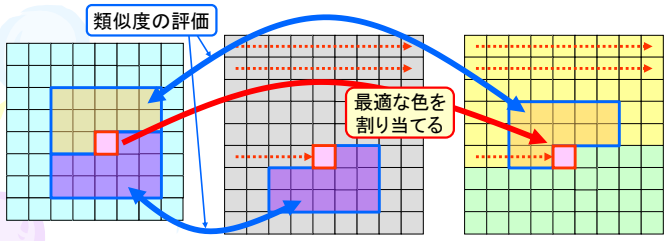
- **M. Ashikhmin (2001)** による**ユーザ制御アルゴリズム**
  - ユーザが与えた**目的テクスチャ**により出力テクスチャを制御する.
  - 候補探索アルゴリズムと同論文内で提案された.
- **M. Ashikhmin, "Synthesizing Natural Textures", 2001**

入力テクスチャ　目的テクスチャ　出力テクスチャ

M. Ashikhmin, "Synthesizing Natural Textures", 2001

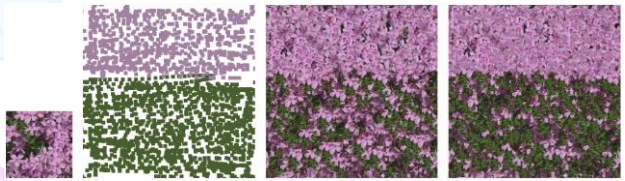
ユーザ制御アルゴリズム：類似度評価

- 類似度評価する近隣ピクセル群
  - 入力テクスチャ：**正方領域**
  - 出力テクスチャ：**上L型領域** → 入力テクスチャの**上L型領域**と比較
  - 目的テクスチャ：**下L型領域** → 入力テクスチャの**下L型領域**と比較



ユーザ制御アルゴリズム：反復処理

- 出力テクスチャを目的テクスチャに近づけるため，**合成処理を反復**する.
- 反復ごとに，出力テクスチャの各ピクセルに対して，**前回反復で割り当てた入力テクスチャのピクセルを利用して候補ピクセルを決定**する.
- 以下，入力テクスチャのピクセルを**入力ピクセル**，出力テクスチャのピクセルを**出力ピクセル**と呼ぶ.

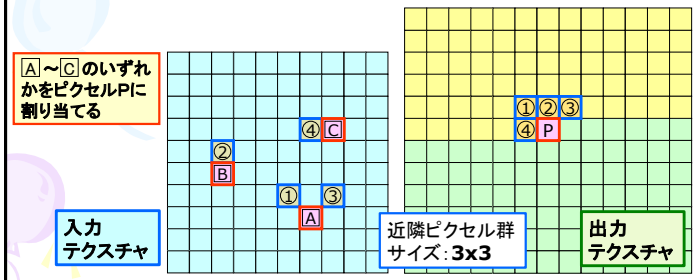


入力テクスチャ　目的テクスチャ　出力テクスチャ 反復1回　出力テクスチャ 反復5回

M. Ashikhmin, "Synthesizing Natural Textures", 2001

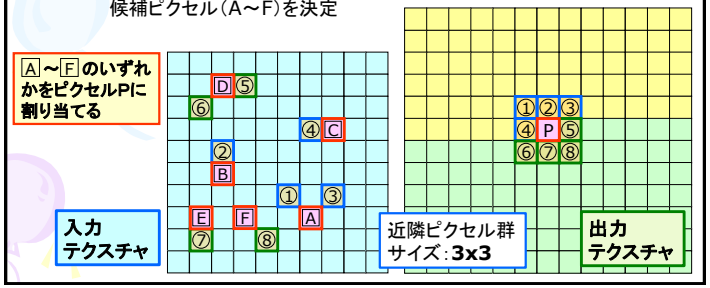
ユーザ制御アルゴリズム：反復処理

- 1回目の反復
  - 各出力ピクセルに割り当てた入力ピクセルの座標を記憶する座標記憶用配列を**ランダムな座標**で**初期化**
  - 処理中の出力ピクセルPに対して，**上L型領域**の各ピクセルに割り当てた入力ピクセル(①～④)を用いて候補ピクセル(A～C)を決定



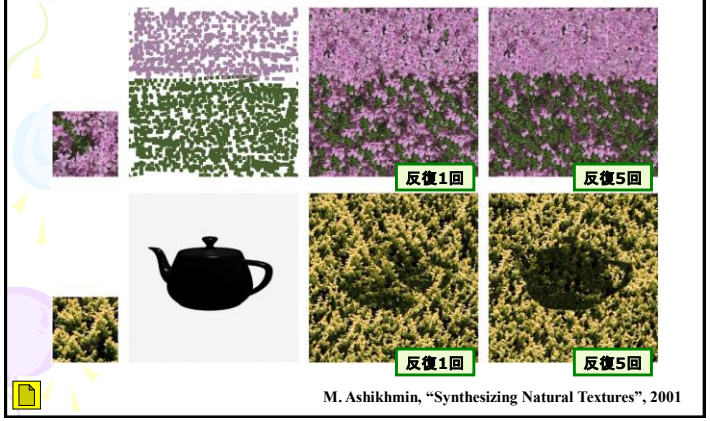
ユーザ制御アルゴリズム：反復処理

- 2回目以降の反復
  - 各出力ピクセルに**前回反復で割り当てた(座標記憶用配列に記憶されている)入力ピクセル**を利用して候補ピクセルを決定
  - 処理中の出力ピクセルPに対して，**現在の反復**で**上L型領域**の各ピクセルに割り当てた入力ピクセル(①～④)，および，**前回反復**で**下L型領域**の各ピクセルに割り当てた入力ピクセル(⑤～⑧)を用いて候補ピクセル(A～F)を決定



ユーザ制御アルゴリズム：反復処理

- 実験例



ユーザ制御アルゴリズム：反復処理

- 実験例