GPUを用いたインタラクティブフォトモザイク

Interactive Photomosaic Using GPU

藤澤　誠†,　加藤博一†

Makoto Fujisawa† and Hirokazu Kato†

Abstract　A photomosaic is a type of decorative art made up from various other photographs. We present a method for quickly generating photomosaics and propose an interactive recursive photomosaic system. Users can operate the system by using a wall display with a touch input function, which allows them to alter the appearance of the image dynamically.

キーワード：フォトモザイク，GPU，ウォールディスプレイ

1. まえがき

ユーザへの情報提示として用いられるディスプレイは，フルハイビジョンなどに対応した高解像度化が進んでおり，今後，4kやスーパーハイビジョンへとさらなる高精細化が勧められると考えられ，そのためのコンテンツも高解像度を生かしたものが必要である．本研究では，高解像度を生かしたコンテンツとしてフォトモザイクに注目し，高解像度のフォトモザイクをGPUを用いた並列化によりリアルタイムに生成する手法を提案する．さらに，タッチパネル機能付きのウォールディスプレイを用いたインタラクティブフォトモザイクを提案する．

フォトモザイクは，写真を要素としたモザイク画の一種である．元の画像をタイルと呼ばれる小さな領域で分割し，そこにデータベースに格納された多くの写真 (タイル画像) を元の画像の色情報に基づき当てはめることで新しい画像を生成する．高解像度ディスプレイであれば，フォトモザイクを表示したときに一つ一つのタイル画像もある程度判別可能であり，インタラクティブフォトモザイクではそれを拡大して見ることができるようにする．さらに，絵の中にさらなる絵が隠されているというフォトモザイクの再帰性を拡張し，タイルを構成する画像もまたフォトモザイクであるという特徴を持たせる．これにより，例えば，写真間の接続関係を利用し，人と人，人と物，物と物との関係性などを表現することも可能となる．

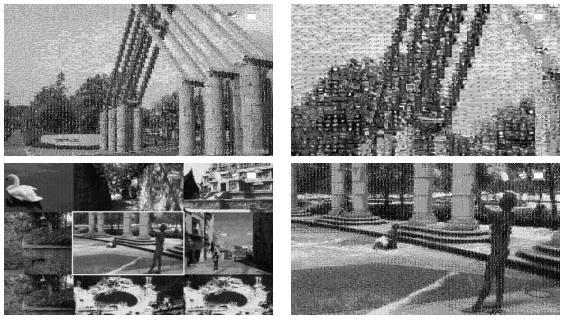

図 1　再帰的なフォトモザイクによるズーム処理
Scaling up by the recursive photomosaic.

2. 関連研究

グリッド状にタイルを配置するフォトモザイクの生成手法として，Tran[5] はタイルとそれを配置するグリッドの全ピクセルの色を比較し，差が最小のものを検索することで元の画像に類似したフォトモザイクを生成した．しかし，この手法では，タイルの全ピクセルを参照する必要があり，リアルタイムでの計算には不向きである．より高速なフォトモザイクの生成のために，Di Blasi ら [2] はタイル画像を9分割 (3×3) し，各分割領域の平均色を計算することで得られる各画像の特徴を表す27次元のベクトルを用いた．また，フォトモザイクを他のアプリケーションに応用する研究としては，動画ナビゲーション [3] や動画のフォトモザイク [4] などがあげられる．

3. フォトモザイク

本研究で用いるフォトモザイクは，元の画像 (サイズ $w_o \times h_o$) を格子状に分割し，各格子 (タイル) 領域 (サイズ $w_t \times h_t$) の色特徴ベクトルから類似画像を検索し，あてはめていくことで合成される．色特徴ベクトルには

2010年3月30日受付，2010年6月23日再受付，2010年6月28日採録
†奈良先端科学技術大学院大学
（〒 630–0192 奈良県生駒市高山町 8916–5, Tel 0743-72-5332）
†Nara Institute of Science and Technology
(8916–5 Takayama-cho, Ikoma-shi, Nara, 630–0192, Japan)

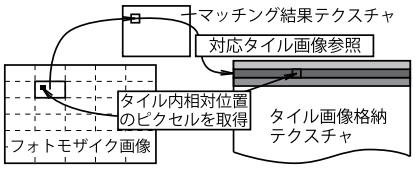

図 2 フォトモザイク合成
Synthesis of photomosaic.

タイルを d^2 分割 (d×d) したときの各分割領域の平均色 $T_r^{ij}, T_g^{ij}, T_b^{ij}$ $(i,j = 0, ..., d)$ からなる $3d^2$ 次元ベクトルを用いる．

フォトモザイク生成手順は，1) 前処理として，データベースに登録された画像の色特徴ベクトルを計算，さらに，タイルの大きさがあらかじめ決まっている場合，画像をタイルの大きさにリサイズ，2) 元画像を読み込み，各グリッドの色特徴ベクトルを計算し，前処理で構築されたツリーを用いて，最近傍探索により類似画像を取得，3)GPU を用いた並列処理によりフォトモザイク画像を合成，である．

3.1 前処理

タイルとして用いる画像群に対して，4 分割した各領域の平均色からなる色特徴ベクトル $(T_r^{ij}, T_g^{ij}, T_b^{ij}$ $(i,j = 0, ..., d))$ を計算する．そして，色特徴ベクトルの値は近傍探索のために，Approximate Nearest Neighbor(ANN) KD 木 [1] に格納する．フォトモザイクでは類似画像の完全性は必要でないことから，近似的なアルゴリズムではあるが，高次元においても高速に処理可能な近傍探索手法である ANN を用いた．また，画像をタイルの大きさ (w_t, h_t) にリサイズし，各リサイズ画像を 1 次元配列に再配置することで，データベース内のすべての画像を 1 枚の 2 次元テクスチャとして GPU に転送する，

3.2 マッチング画像探索

タイル画像の色特徴ベクトル場から元画像とのマッチングを行う．ここで，元画像はタイルサイズに対して縦横ともに整数倍の大きさであるとする．読み込まれた元画像を $n \times m$ に分割し，各分割領域における色特徴ベクトルを計算し，3.1 節で述べた ANN を用いて類似画像を探索する．ANN のようなツリー型探索は，GPU 上での実行に適していないため，画像マッチングは CPU 上で行う．ANN を用いた場合，CPU 上でもインタラクティブに実行するための充分な速度が得られる．また，なるべく多くの種類の画像をモザイクに用いたいため，最近傍の一つだけでなく近傍 N 個の画像を探索し，その中からランダムに画像を決定する．4 章で述べるようにタイルをユーザが選択できるようにするために，マッチング結果は保持しておく．

3.3 GPU によるモザイク画像生成

マッチング結果とリサイズされたタイル画像からフォトモザイクを生成する．CPU 側でフォトモザイクを生成すると，表示のために GPU に転送する必要があり，特に高解像度の場合，転送時間がボトルネックとなる．そのため，生成された画像を GPU に転送するのではなく，GPU 内でモザイク画を合成することで，データを CPU から毎回転送する必要性がなくなり，かつ，直接描画用テクスチャとして使用することで拡大処理もスムーズに行える．

GPU のピクセルシェーダを用いてフォトモザイク生成を行う．シェーダは複数存在し (NVIDIA GeForce GTX285 で最大 240 個)，描画画面上の各ピクセルごとに並列に処理することができる．そして，メインメモリーに格納されたマッチング結果を $n \times m$ の大きさのテクスチャデータとして転送することで，ピクセルシェーダに値を渡す．マッチング結果は $[1, N]$ の自然数であるため，OpenGL 拡張である `GL_EXT_texture_integer` を用いて，整数値を変換なしに直接転送する．リサイズされたタイル画像群を格納したテクスチャは，前処理の段階で GPU に転送しているため，モザイク生成ステップではマッチング結果のみを受け取る．マッチング結果は分割数と同じピクセル数の画像として転送されるため，その転送コストは画像そのものを転送するのに比べて非常に小さい．

図 2 にピクセルシェーダを用いたフォトモザイク画像の合成手法を示す．ピクセルシェーダはピクセルごとに処理されるため，モザイクを行う元の画像と同じ大きさのビューポートを設定し，矩形ポリゴンを描画することで対応するテクスチャ座標から元画像上のピクセル位置 (i,j) を得る．(i,j) をタイル座標 $(t_x, t_y) = (\mathrm{floor}(i/w_t), \mathrm{floor}(j/h_t))$ に変換し，マッチング結果を格納したテクスチャの (t_x, t_y) の値を参照することで対応する画像の番号を取得する．ここで，floor は小数点以下切り捨てを表す．また，タイル内のローカルピクセル位置 $(t_i, t_j) = (i - t_x w_t, j - t_y h_t)$ からタイル画像格納テクスチャの x 座標値を計算し，最終的なピクセル色を得る．

4. 再帰的なフォトモザイク

高解像度フォトモザイクをナビゲーションなどに応用するために，再帰的なフォトモザイクアプリケーションを提案する．このアプリケーションでは，高解像度ディスプレイに表示されたフォトモザイクから，ユーザが任意の画像を選択する．高解像度ディスプレイにより各タイル画像をユーザはある程度判別することが可能である．選択された画像を元に，新たにフォトモザイクをダイナミックに生成し，画像を最大化する．このとき，ズームするようなエフェクトをかけることで，ユーザはあたかもすべてのタイル画像がフォトモザイクで構成されている感覚を得る．このとき，拡大するときに周囲 1 周りの画像も含めてフォトモザイクを生成しておくことで，拡大時の不自然さを軽減する．そして，ウォール型ディスプレイにより直感的なタッチ操作を提供する．

5. 結　果

インタラクティブフォトモザイクを実装し，ウォールディ

図 3 ウォールディスプレイを用いた画像選択
Picking up the tile image using wall display.

スプレイとしてタッチパネル機能を持つ 250 インチのディスプレイ (3 台のプロジェクタを使用) を用いて実験した．実験では 2800×1050 の解像度の表示領域にハイビジョン解像度 (1920×1080) の画像を描画した．**図 3** はインタラクティブなフォトモザイクの操作の様子を示している．フォトモザイク中のタイルを触ることにより，そのタイルが拡大され，再帰的にフォトモザイクが描画される．フォトモザイクからフォトモザイクへの拡大処理の間のフレームを**図 1** に示す．図 3 の実験では被験者にインタラクティブフォトモザイクを体験してもらったが，このとき，処理時間が数秒程度かかると選択されているか不安になり，何度もタッチするシーンが見られた．このことから処理時間は少なくとも 1 秒程度以下にするべきである．

CPU と GPU による計算時間の比較結果を**表 1** に示す．CPU に Intel Core 2 Duo 3.16GHz, GPU として NVIDIA GeForce GTX 285 (ビデオメモリー 1GB) を搭載した PC 上で実行した．画像を選択してから拡大を始めるまでの処理時間を 10 回計測し，その平均を表 1 に示した．この処理時間には ANN によるマッチングも含まれている．拡大するときに周囲の画像も含めてフォトモザイクを生成するため，合計 9 枚のマッチング，モザイク画像生成にかかった時間である．解像度が低い場合は，計算時間に対する画像の読込み時間の割合が大きく GPU 処理のアドバンテージは小さい．しかし，HD を超える解像度になると画像のテクスチャへの転送時間の割合が大きくなり，転送すべき情報が少なくすむ GPU 処理と CPU 処理の差が大きくなっている．また，GPU で 1920×1080 以上の画像の処理速度がほとんど変わらないのは，スレッド数が増えたことで，メモリーレイテンシが小さくなったためと考えられる．

また，色特徴ベクトル算出におけるタイルの分割数 d は，生成されるフォトモザイク画の元の画像に対する再現度と計算速度のバランスから決定した．**図 4** に $d = 1, 2, 3, 4$ の時のフォトモザイク画の比較を示す．$d = 1$ では再現度が明らかに低いが，$d = 2$ 以上であれば問題ない．また，計算時間はベクトル次元数に比例して大きくなることを実験により確認しており，そのため $d = 2$ とした．

6. むすび

本研究では，タッチパネル機能付きのウォールディスプレイを用いたインタラクティブフォトモザイクを提案した．高解像度ディスプレイにおける視聴距離に対して，印象が変化するコンテンツとしてフォトモザイクに注目し，GPU を用いることで高解像度のフォトモザイクを高速に生成する手法を開発した．しかしながら，周囲画像まで含めてリアルタイムで生成するためには，さらなる高速化が必要である．今後の研究としては，フォトモザイクの写真間の接続関係を利用し，画像間の関係性の表現や観光ナビゲーションなどに応用するなどがあげられる．

表 1 画像選択から拡大までの処理時間 (秒)
Photomosaic generation times.

解像度	640×480	1280×1024	1920×1080	3840×2160
CPU	0.16	0.55	0.86	2.27
GPU	0.15	0.37	0.54	0.55

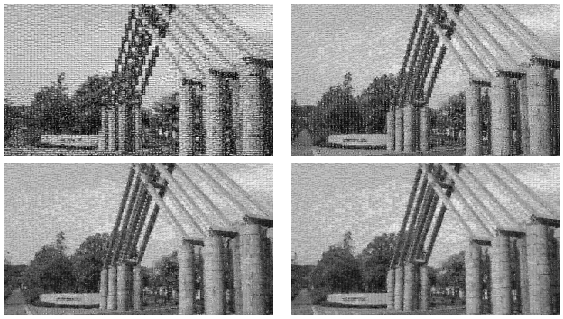

図 4 分割数 d の違いによるクオリティの変化 (左上:$d = 1$, 右上:$d = 2$, 左下:$d = 3$, 右下:$d = 4$)
Difference in quality by d.

〔文　献〕

1) S. Arya, D. Mount, N. Netanyahu, R. Silverman and A. Wu: "An optimal algorithm for approximate nearest neighbor searching fixed dimensions", Journal of the ACM, 45(6), pp.891-923 (1998)
2) G. Di Blasi, G. Gallo and M. Petralia: "Smart ideas for photomosaic rendering", In Proc. Eurographics Italian Chapter Conference (2006)
3) K. Kim, I. Essa and G. Abowd: "Interactive mosaic generation for video navigation", In Proc. MULTIMEDIA '06, pp.655-658 (2006)
4) A. Klein, T. Grant, A. Finkelstein and M. Cohen: "Video mosaics", In Proc. NPAR 2002, pp.21-28 (2002)
5) N. Tran: "Generating photomosaics: an empirical study", In Proc SAC '99, pp.105-109 (1999)

藤澤　誠 (ふじさわ　まこと) 2003 年静岡大学工学部機械工学科卒業．2005 年同大学大学院理工学研究科修士課程修了．2008 年同博士課程修了 (博士 (工学))．同年，奈良先端科学技術大学院大学助教．物理シミュレーション等の研究に従事．

加藤　博一 (かとう　ひろかず) 1986 年大阪大学基礎工学部制御工学科卒業．1988 年同大大学院修士課程修了．1989 年同大学基礎工学部助手．1996 年講師．1998 年ワシントン大学客員研究員．1999 年広島市立大学情報科学部助教授．2003 年大阪大学大学院基礎工学研究科助教授．2007 年奈良先端科学技術大学院大学情報科学研究科教授．博士 (工学)．拡張現実感，ヒューマンインタフェースの研究に従事．