RI-001

低重要度の行と列の削除と挿入による画像リサイジング

Image Resizing by Deletion and Insertion of Least Important Rows and Columns

傅 暁宇† 井上 光平† 浦浜 喜一†
Xiaoyu Fu Kohei Inoue Kiichi Urahama

あらまし

重要度の低い行や列を削除・挿入して画像のアスペクト比やビデオの長さを変える手法を提案し，画像やビデオのデータ圧縮や写真の構図の改善に応用する．

1. まえがき

重要な物体のサイズを保って画像のアスペクト比を変えるリサイジング (resizing) 技法が発表されてきている．なかでも Avidan らのシームカービング (seam carving: SC) 法 [1] はよく知られており，画像のデータ圧縮へも応用されている [2, 3]．SC 法では，重要度の低い任意曲線を逐次に削除・挿入して画像サイズを変える．Cho ら [4] は，SC 法で画像を縮小する場合について重要度を拡散させる手法を提案し，削除する曲線を水平線 (行) と垂直線 (列) だけに限定した簡便法も示した．

SC 法は動的計画法を使っているが多くの計算を要し，ビデオの 3 次元時空間画像に適用すると計算時間が問題になる．そこで本稿では，Cho ら [4] と同様に行と列だけを扱う計算量が少ない手法を考える．Cho らは画像を縮小する場合だけを示したが，本稿では画像拡大やビデオの時間短縮にも拡張する．また，SC 法や Cho らの手法では各画素の重要度として勾配強度を用いているが，本稿では必要最小計算量の重要度として行結合度と列結合度，加えてビデオでは時間結合度を定義し，それに基づいて画像の行や列，ビデオのフレームを削除・挿入する手法を提案する．

また，このような行と列の増減によるリサイジング法を画像やビデオのデータ圧縮に応用し，画像圧縮では勾配強度による画像縮小が有効であり，Achanta らの重要度 [5] で圧縮すると復元誤差が更に小さくなることを示す．続いて，行と列の増減を利用して，画像サイズは変えずに写真の構図を改善する手法を提案し，この場合にも Achanta らの重要度が有用であることを示す．

2. 勾配強度に基づくリサイジング

まず最初に，計算量が少ない重要度として，サイズを変更する方向の勾配強度だけを用いる手法を提案する．

2.1 結合度

画像中の隣接行や隣接列，ビデオの隣接フレーム間の結合の強さを定義する．本提案法では結合度が低い画像中の行や列，ビデオのフレームを逐次に削除・拡大して，横幅 L，高さ M の画像を横幅 L'，高さ M' に変える．ビデオでは合わせてフレーム数 N を N' に変える．

2.1.1 列結合度

画像 (ビデオではフレーム) の第 i 列の列結合度を

$$c_{xi} = \sum_{j=1}^{M} |g_{xij}| \tag{1}$$

と定義する．ここで g_{xij} は 画素 (i,j) での横 (x) 方向の勾配であり，ソーベルフィルタで求める．この c_{xi} は，通常重要度と呼ばれているものと同じであるが，ここで“結合度”という呼び方をした理由は，隣と強く結合している画像の行や列，ビデオのフレームを削除したり，その間に行や列，フレームを挿入するのは好ましくない，という意味合いを表すためである．c_{xi} の値が大きい列は隣の列との結合が強い．

2.1.2 行結合度

同様に，第 j 行の行結合度を

$$c_{yj} = \sum_{i=1}^{L} |g_{yij}| \tag{2}$$

と定義する．ここで g_{yij} は 画素 (i,j) での縦 (y) 方向の勾配である．

2.1.3 時間結合度

ビデオの第 k フレームの (i,j) 画素での時間 (t) 方向の勾配を g_{tijk} とすると，第 k フレームの時間結合度を

$$c_{tk} = \sum_{i=1}^{L}\sum_{j=1}^{M} |g_{tijk}| \tag{3}$$

と定義する．

2.2 リサイジング

以上の結合度が大きい行や列，フレームは隣の行や列，フレームとの結合が強い，すなわち画素値の変化が激しい．そのような行や列，フレームを削除したり，そのような箇所に新たに行や列，フレームを挿入すると画素値に不連続が生じて画質が劣化する．従って，結合度が最も小さい行や列，フレームを削除・追加することにする．

しかし，結合度が小さい行や列，フレームを逐次に削除すると，連続した行や列，フレームが削除され，やはり不連続が生じてしまう．従って，それらが飛び飛びに

†九州大学大学院芸術工学研究院

削除・挿入されるように，Cho らの手法 [4] と同様に，削除・挿入の度に結合度を増加させる．この重要度可変法は，サイズ縮小のときには拡散と解釈できる [4] が，拡大のときには拡散とは無関係である．

なお，SC 法や Cho らの手法 [4] では横幅を変えるときも高さを変えるときも勾配強度 $|g_{xij}|+|g_{yij}|$ を用いているが，サイズ変更の向きと直交する勾配は歪みの要因にもなり，計算量の点でも不要であるので，本章では上記の結合度を用いる．

2.2.1 画像の横幅縮小

横幅 L を $L'(<L)$ に減らす場合の手順は以下の通りである．

Step 1) 各列の列結合度 c_{xi} を計算する．
Step 2) c_{xi} が最小の列 i を求める．
Step 3) その列の左隣の列 $i-1$ と右隣の列 $i+1$ の列結合度 $c_{x,i-1}$ と $c_{x,i+1}$ を次の $c'_{x,i-1}$ と $c'_{x,i+1}$ に更新する (α は例えば 0.5)．

$$c'_{x,i-1} = c_{x,i-1} + \alpha c_{xi} \quad (4)$$

$$c'_{x,i+1} = c_{x,i+1} + \alpha c_{xi} \quad (5)$$

Step 4) 第 i 列を削除して，第 $i+1$ 列め以降を左に 1 列詰める．
Step 5) 横幅が L' ならば終了．そうでなければ Step 2 へ戻る．

なお，この手順の間，c_{xi} は Step 1 で求めた値を使う．

2.2.2 画像の横幅拡大

横幅 L を $L'(>L)$ に増やす場合の手順は以下の通りである．

Step 1) 各列の列結合度 c_{xi} を計算する．
Step 2) c_{xi} が最小の列 i を求める．
Step 3) その列の左隣の列 $i-1$ と右隣の列 $i+1$ の列結合度 $c_{x,i-1}$ と $c_{x,i+1}$ が $c_{x,i-1} < c_{x,i+1}$ ならば step 4 へ．そうでなければ step 6 へ．
Step 4) 列 $i-1$ と列 i の列結合度 $c_{x,i-1}$ と c_{xi} を次の $c'_{x,i-1}$ と c'_{xi} に更新する (β は例えば 1.5)．

$$c'_{x,i-1} = \beta c_{x,i-1} \quad (6)$$

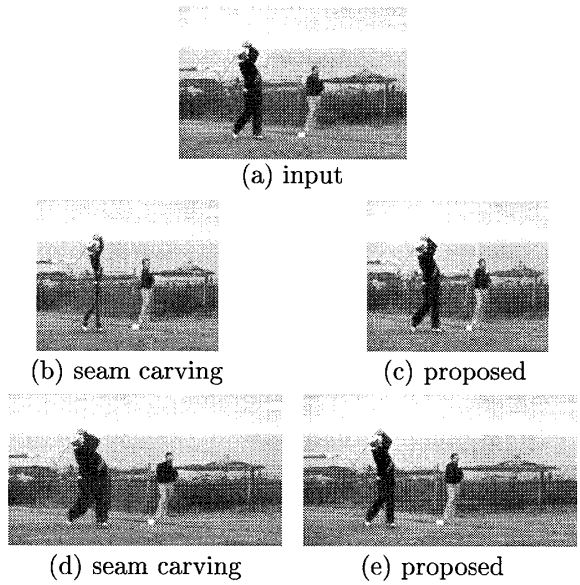
$$c'_{xi} = \beta c_{xi} \quad (7)$$

Step 5) 列 $i-1$ と列 i の間に 1 列挿入する．挿入列の画素値は列 $i-1$ と列 i の平均値とする．挿入列の列重要度は $\beta(c_{x,i-1}+c_{xi})/2$ とする．
Step 6) 列 i と列 $i+1$ の列結合度 c_{xi} と $c_{x,i+1}$ を次の c'_{xi} と $c'_{x,i+1}$ に更新する．

$$c'_{xi} = \beta c_{xi} \quad (8)$$

$$c'_{x,i+1} = \beta c_{x,i+1} \quad (9)$$

Step 7) 列 i と列 $i+1$ の間に 1 列挿入する．挿入列の画素値は列 i と列 $i+1$ の平均値とする．挿入列の列重要度は $\beta(c_{xi}+c_{x,i+1})/2$ とする．
Step 8) 横幅が L' ならば終了．そうでなければ Step 2 へ戻る．



(a) input
(b) seam carving
(c) proposed
(d) seam carving
(e) proposed

図 1: 横幅の拡大・縮小の例

2.2.3 画像の高さの拡大・縮小

画像の高さ (縦幅) を拡大・縮小する手順も，列結合度の代わりに行結合度を用いるだけで以上と同じである．

2.2.4 ビデオのフレーム数の増加減少

同様に，ビデオのフレーム数を増加・減少させる手順も，列結合度の代わりに時間結合度を用いるだけで以上と同じである．フレームサイズとフレーム数の両方とも変える場合には，計算時間の節約のため，まずフレーム数を変えてから各フレームのサイズを変える．

なお，Achanta らの重要度はビデオには拡張されていないので，現状ではビデオの重要度は本稿のように画素値の勾配に基づいて計算するしかない．

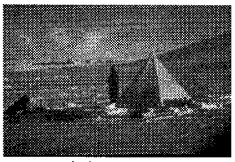
2.3 画像リサイジングの実験例

SC 法と本提案法とを比べるために，R.Achanta のウェブページ [6] に掲載されている画像について実験した．SC 法の結果が良好な画像例では本提案法も同じように良好な結果が得られたが，SC 法で歪が生じる場合でも本提案法では概ね良好な結果が得られた．そのような例を以下に示す．

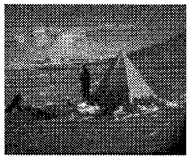
図 1(a) の画像の横幅を 481 から 384 に縮小した結果を (b) と (c) に示す．(b) は SC 法，(c) が提案法である．図 1(b) では人物の横幅が縮んでしまっているが，(c) ではほぼ原形が保たれている．

図 1(d) と (e) は横幅を 577 に広げた場合であり，SC 法 (図 1(d)) では人物の横幅も広がってしまっているが，提案法 (図 1(e)) ではほぼ原形のまま保たれている．

計算時間は図 1(c) が 0.02 秒，図 1(e) が 0.1 秒であった．計算環境は Intel Core 2 Duo CPU 2.93GHz，4GB メモリ，Windows 7, C++言語である．


(a) input


(b) seam carving


(c) proposed


(d) seam carving

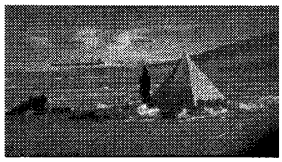
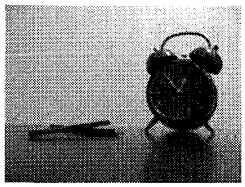
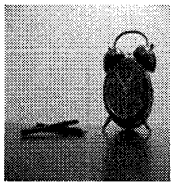

(e) proposed

図 2: 横幅の拡大・縮小の例


(a) input


(b) linear


(c) seam carving

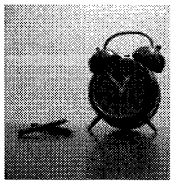
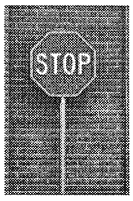

(d) proposed

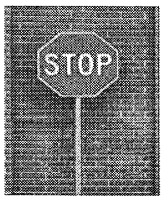
図 3: 横幅縮小の例

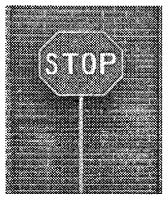
図 2 は別の画像例であり，SC 法では人物の横幅が変わったり，テントの形状が歪んでしまっているが，提案法ではほぼ原形のまま保たれている．

他の例を図 3 と図 4 に示す．図 3 は (a) の画像の横幅を 500 から 350 に縮めた結果であり，(b) は線形縮小，(c) が SC 法，(d) が提案法である．(b) では時計の横幅が縮んでいるが，(d) ではほぼ丸いまま保たれている．但し，鉛筆の長さは (d) では (b) よりも少し短くなっている．図 4 は，(a) の画像の横幅を 394 から 490 に広げた場合であり，(b) は線形拡大，(c) が SC 法，(d) が提案法である．(c) では標識の横幅が (b) よりも広がってしまっているのに対し，(d) では広がりは僅かである．

なお，Achanta らの重要度を用いても，図 1(c),(e)，図 2(c),(e)，図 3(d)，図 4(d) とほぼ同じ結果が得られたが，計算量は上記の結合度を用いるほうが少なく，高速である．


(a) input


(b) linear


(c) seam carving

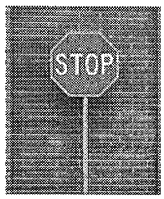

(d) proposed

図 4: 横幅拡大の例

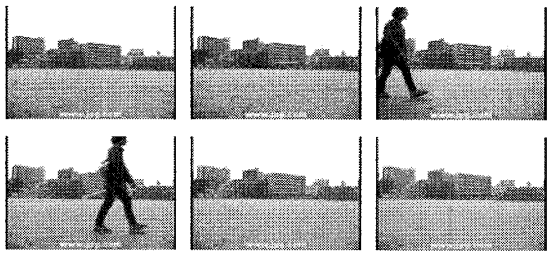


図 5: 入力ビデオ

2.4 ビデオのリサイジング

ビデオのフレーム数を減らす場合，線形な短縮すなわち早送りでは重要な物体の動きが速くなりすぎて視認しづらくなる場合がある．そこで，動きが少ない時間帯を主に短縮して，主要物体が動く時間帯はできるだけ縮めないのが望ましい．これは静止画像のリサイジングでの横幅を時間に置き換えたものであり，2.2.1 節の手法がそのまま使える．

フレーム数に加えてフレームサイズも縮める場合には，2.2.4 節で述べたように，まずフレーム数を変えてから次に各フレームのサイズを変えるが，各フレームを別々に以上の手順で縮小・拡大すると映像の時間的なブレが生じるので，時間的な連続性を保つために以下の手順で行う．

Step 1) フレームの時間結合度に基づいて 2.2.1 節と同じ手順でフレーム数を減らす．

Step 2) 短縮ビデオの各画素での勾配 g_{xij} と g_{yij} を求め，時間方向に平滑化してから行と列の結合度を求める．

Step 3) 平滑化した行結合度と列結合度に基づいて，短縮ビデオの第 1 フレームの縦横を縮小する．

Step 4) 第 2 フレーム以降では，直前のフレームで削除された列や行と同じ行や列の結合度を $\gamma(<1)$ 倍してから 2.2.1 節の手順で縦横を縮小する (γ は例えば 0.7)．

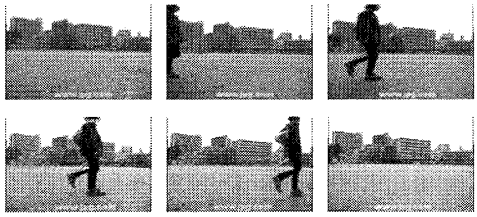
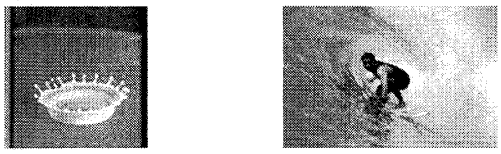


図 6: リサイジングビデオ



(a) milkdrop (512 × 512)　(b) surfing (640 × 426)

図 7: 送信画像

2.4.1 ビデオリサイジングの実験例

フレームサイズ 720 × 480，フレーム数 180 のビデオを，フレームサイズ 600 × 400，フレーム数 90 に縮小した．縮小前のビデオを図 5 に，縮小後を図 6 に示す．どちらも左上が第 1 フレーム，右下が最終フレームで，中間の 4 枚は時間を 5 等分したものである．図 5 の入力ビデオでは途中から人が左からフレームインし，右に歩いてフレームアウトした後，静止背景が続く．これを時間短縮した図 6 のビデオでは前後の静止時間が短縮され，人がフレームインする時刻が早くなり，フレームアウトする時刻が遅くなっており，人が写っている時間帯は入力ビデオからあまり縮んでいない．従って，人の動きは早送りされずに，不要な静止区間だけが短縮されている．

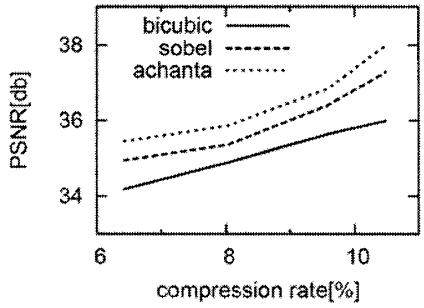
リサイジングの計算時間は 6.14 秒であった．

ビデオのブレの度合いの指標として，各画素での時間方向の勾配強度 $|g_{tijk}|$ の平均値を比較した．フレームサイズは変えずにフレーム数だけを減らしたときのブレ度合いは 7.65 であり，その短縮ビデオの各フレームを別々に縮小したときは 12.83，上記の手順での縮小ビデオのブレ度合いは 9.03 であり，上記手順中の連続性を保つ付加処理の有効性が確認できた．

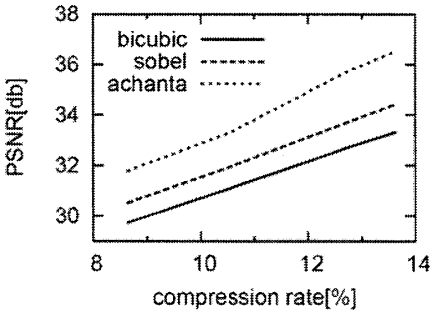
3. 画像やビデオのデータ圧縮への応用

次に，以上のリサイジング法をデータ圧縮に利用する．SC 法の画像圧縮への利用法 [2, 3] では，削除した曲線の場所の情報伝達が煩雑であるが，行と列だと少ない情報で済む．

ここで考える手法は，送信側では，まず行と列を削除して画像サイズを縮小してから JPEG 圧縮し，削除した行と列の場所の情報と JPEG 画像とを送信する．受信側では，受け取った JPEG 画像に削除された行と列を挿入して元のサイズに戻す．挿入する行や列は両隣の行や列の平均値とする．

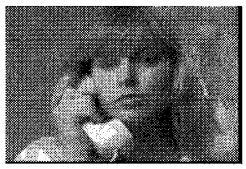


(a) milkdrop



(b) surfing

図 8: 受信画像の PSNR



(a) telephone

(b) football

図 9: 送信ビデオ

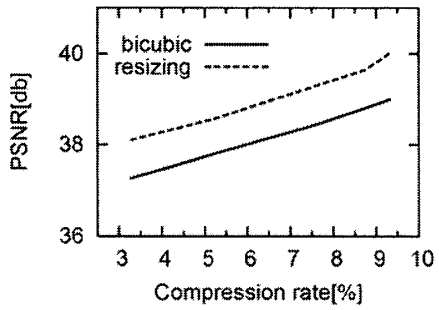
3.1 静止画での実験

図 7 の画像での受信画像の PSNR を図 8 に示す．横軸の圧縮率はここでは “圧縮後のファイルサイズ/原画像のファイルサイズ”×100%である．ここで圧縮後のファイルは “JPEG 画像＋削除行と列の場所情報”である．実線は画像サイズを線形縮小してバイキュービック補間で復元した場合であり，破線は勾配強度によるリサイジング，点線は Achanta らの重要度によるリサイジングである．このように，画像圧縮への応用においては勾配強度よりも Achanta の重要度を用いるほうが復元誤差が小さかった．

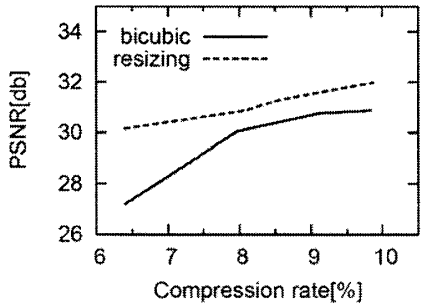
Achanta の重要度の計算時間は図 7(a) が 0.14 秒，図 7(b) が 0.16 秒であった．

3.2 ビデオでの実験

このように Achanta の重要度は画像圧縮に有用であり，高速に計算できるが，ビデオには拡張されておらず，画像に対するアルゴリズムを全フレームに適用すると計算時間がかかるので，勾配強度を用いてビデオを圧縮した．手順は



(a) telephone



(b) football

図 10: 受信ビデオの PSNR

(送信側)
1) ビデオをフレームに分解して，各フレームのサイズを縮小する．
2) 縮小フレームの動画を wmv 形式で圧縮する．
3) 各フレームの削除行と列の情報と wmv 形式動画とを送信する．
(受信側)
1) 受け取った wmv 動画をフレームに分解する．
2) 各フレームでの削除された行と列を挿入する．

図 9 に示すフレームサイズ 720 × 486，フレームレート 30fps，長さ 8 秒のビデオでの PSNR を図 10 に示す．線形縮小してバイキュービック補間したとき (実線) よりも提案法 (破線) のほうが復元誤差が小さい．Achanta の重要度を用いるほうが PSNR が大きいと思われるが，図 9 の各ビデオで，全フレームに Achanta のアルゴリズムを適用すると 41 秒かかる．

図 10 での計算時間は圧縮が 8.19 秒，復元が 2.43 秒であった．

なお，通常のようにフレームを縮小せずに単に wmv 形式に変換したときの圧縮率は，図 9(a) が 11.54%，図 9(b) が 14.81%であり，図 10 の圧縮率はそれよりも小さく，リサイジングの効果が認められる．

4. 写真の構図改善への応用

以上では，行と列の増減をそのままリサイジングに利用したが，次に画像のサイズは変えずに主要物体の位置だけを動かして写真の構図を改善する技法を提案する．写真の構図には種々の基本パターンがあるが，ここでは三分割法をとりあげ，単純な画像例について構図改善法を示す．

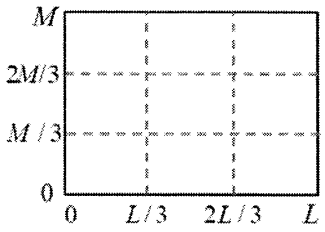


図 11: 三分割点

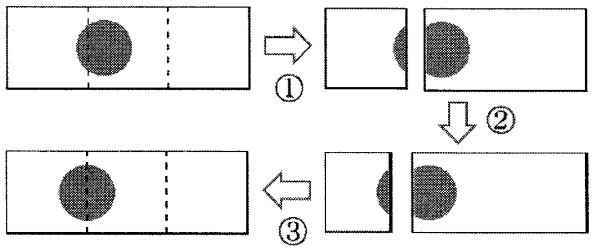


図 12: 主要物体の移動

4.1 三分割法

よく用いられる写真構図である三分割法では，写真の横を三分割する 2 本の垂直線と縦を三分割する 2 本の水平線の上やそれらの交点に主要な被写体を配置する撮影法である．

画像の横幅を L，高さを M とすると，図 11 のように三分割点の x 座標は $L/3$ と $2L/3$，y 座標は $M/3$ と $2M/3$ であり，4 個の交点がある．

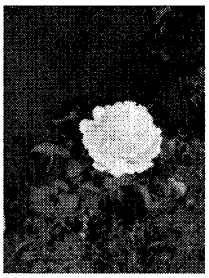
4.2 主要物体の抽出

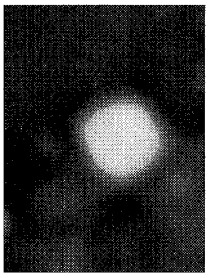
最も簡単な場合として，主要な物体一つだけを三分割点の一つに移動させる場合を考える．

主要物体を抽出するには勾配強度だけでは不十分で，視覚認知に対応した重要度検出法が必要である．ここでは Achanta の手法 [5] による重要度画像をぼかしてモード値を求め，そこを主要被写体の重心位置とする． この物体重心を，そこから最も近い三分割点に動かす．重心位置を (x, y) とすると，例えば $L/3 < x < L/2, M/3 < y < M/2$ であったとする．この場合，(x, y) を $(L/3, M/3)$ に移動させる．そのためには，横方向の操作として，$[0, x]$ の範囲から $x - L/3$ 列を削除し，$[x, L]$ の範囲に $x - L/3$ 列を挿入し，縦方向には $[0, y]$ の範囲から $x - M/3$ 行を削除し，$[y, M]$ の範囲に $x - M/3$ 行を挿入すればよい．

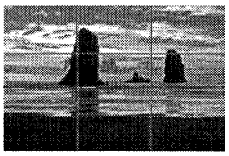
別の言い方をすれば，横方向には，画像を横 $[0, x]$，高さ M の左部分と横 $[x, L]$，高さ M の右部分の 2 枚の画像に分割し (図 12 の矢印 ①)，左部分画像の横幅を $L/3$ に縮小し，右部分画像の横幅を $2L/3$ に拡大する (図 12 の矢印 ②)．それらを繋げて 1 枚の画像にすれば物体中心は $M/3$ の位置になる (図 12 の矢印 ③)．縦方向にも同様に 2 枚に分割してそれぞれの部分画像の高さを拡大・縮小する．これらの拡大・縮小は 2.2.1 節と 2.2.2 節の手


(a) input (612 × 816)

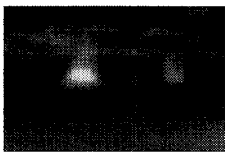

(b) saliency


(c) blurred saliency


(d) output


(e) input (600 × 400)


(f) saliency


(g) blurred saliency

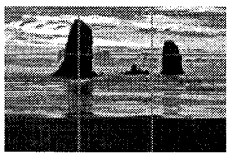

(h) output

図 13: 構図改善の例

順で行う．重心の位置 (x, y) が他の位置でも同様な操作をして最も近い三分割点に移動させる．

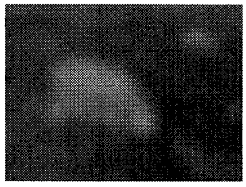
4.3 構図改善の実験例

図 13 に二つの画像例を示す．(a) と (e) が入力画像，(b) と (f) が Achanta の重要度画像 (計算時間は 0.27 秒と 0.14 秒)，それをぼかした画像が (c) と (g) である．これらから画素値のモード点を求めると，1 番目の画像では花の中央，2 番目の画像では左側の岩の下部中央が主要物体の重心として抽出される．それらを三分割点 (それぞれ右上と左上) に移動させた結果が (d) と (h) である．

図 14 は二つの物体 (羊) を移動させた結果である．(a) が入力画像，(b) が Achanta の重要度画像，それをぼかした画像が (c) である．この場合は画像の縦と横を 3 分割して，部分画像を拡大・縮小する．左下の羊を左下の三分割点に，右上の羊を右上の三分割点に移動させた結果が (d) である．上部の柵や羊が変形しているが，このような簡単な手法でもまずまずの結果が得られている．


(a) input (600 × 450)


(b) saliency


(c) blurred saliency


(d) output

図 14: 二つの物体の移動

5. むすび

行と列を削除・挿入する画像リサイジング法として以下のような手法を提案し，応用法を示した．

1) サイズを変える向きだけの勾配を重要度とする手法を提案した．

2) 縮小に加えて拡大についても重要度可変法を提案した．

3) ビデオの短縮にも応用し，ブレを抑制する重要度伝搬法を提案した．

4) 画像とビデオのデータ圧縮に応用し，Achanta の重要度の有用性を示した．

5) Achanta の重要度に基づく写真の構図改善法を提案した．

本稿では重要度として勾配強度だけや Achanta の重要度だけを用いたが，それらを併用した重要度 (例えば両者の和や積) を用いるリサイジング法や，複数の物体を移動させる構図改善法での歪みの低減などが今後の課題である．

参考文献

[1] S. Avidan and A. Shamir, "Seam carving for content-aware image resizing", ACM Trans. Graph., 26, 3, 10, 2007.

[2] N. T. N. Anh, W. Yang and J. Cai, "Seam carving extension: a compression perspective", Proc. ACM MM, pp.825-828, 2009.

[3] 寺田一成，稲積泰宏，堀田裕弘, "Seam carving に基づく空間スケーラブル符号化の検討", 画像符号化 (PCSJ) シンポジウム，PCSJ-2-02, 2009.

[4] S. Cho, H. Choi, Y. Matsushita and S. Lee, "Image retargeting using importance diffusion", Proc. ICIP, 2009.

[5] R. Achanta and S. Susstrunk, "Saliency detection for content-aware image resizing", Proc. ICIP, 2009.

[6] http://ivrg.epfl.ch/supplementary_material/RK_ICIP09/index.html