計測自動制御学会東北支部 第 287 回研究集会 (2014.3.17)
資料番号 287-5

圧縮映像においてブロック単位で欠損した輝度値と動きベクトルの修復

Recovery of Lost Intensity Values and Lost Motion Vectors of Blocks in Compressed Videos

○坂本脩平，阿部正英，川又政征

○ Shuhei Sakamoto, Masahide Abe, Masayuki Kawamata

東北大学

Tohoku University

キーワード： 圧縮映像 (compressed video)，データ欠損 (lost data)，映像修復 (video recovery)，テクスチャ合成 (texture synthesis), エッジ (edge)，動きベクトル (motion vector)

連絡先： 〒 980-8579 仙台市青葉区荒巻字青葉 6–6-05
東北大学 大学院 工学研究科 電子工学専攻 川又研究室
坂本脩平，Tel.: (022)795-7095，Fax.: (022)263-9169，E-mail: shuhei@mk.ecei.tohoku.ac.jp

1. まえがき

災害時など，データ通信が急激に増加する状況では，通信ネットワークにおいて輻輳が発生し，データの一部が欠損する事態が発生する．例えば，携帯端末で映像を受信する際にデータ欠損が発生すると，映像に歪みが生じ，映像から情報を正確に得ることができなくなる．この問題を解消するため，データが欠損した圧縮映像に対する修復手法が必要となる．画像や映像の輝度値や動きベクトルが欠損した場合を想定した修復手法は，今日まで様々な手法が提案されている [1–4]．しかし，いずれの修復手法も，輝度値が欠損した場合もしくは動きベクトルが欠損した場合に限定しており，データ欠損がある圧縮映像の総括的な修復手法は検討されていない．

今日用いられている AVC/H.264 など，多くの映像符号化方式において予測符号化が用いられている．予測符号化では，原映像を I ピクチャと P ピクチャ，B ピクチャに分けて符号化する．I ピクチャは，原フレームの輝度値を用いて符号化される．また，P ピクチャおよび B ピクチャは，参照ピクチャからの動きベクトルと予測誤差を用いて符号化される．したがって，I ピクチャが欠損した場合，欠損した輝度値を修復する必要があり，P ピクチャおよび B ピクチャが欠損した場合，欠損した動きベクトルを修復する必要がある．ここで，予測誤差は復号後の画像に及ぼす影響が少ないため，欠損した予測誤差は修復対象とせず，復号時に用いない．

本稿では，まず，データ欠損がある圧縮映像の総括的な修復手法として，輝度値を修復する輝度値ベースの修復手法と，動きベクトルを修復する動きベクトルベースの修復手法について

それぞれ提案する．次に，人工的に欠損させた動画像を対象とした修復実験を通して，提案する 2 つの修復手法がそれぞれの修復目標を達成していることを示す．最後に，輝度値ベースの修復と動きベクトルベースの修復に要する 1 フレームあたりの計算時間をそれぞれ示す．さらに，DVD-Video 規格に基づく映像を対象とする場合，輝度値ベースの修復と動きベクトルベースの修復を同等のオーダーで計算できることを示す．

2. 輝度値ベースの修復手法

本節では，本稿で提案する輝度値ベースの修復手法について述べる．本稿では，欠損領域に隣接しているエッジを優先的に修復することで，エッジが方向を保存して修復されることを目標とする．この目標を達成するため，文献 [5] の手法における次の 3 つの問題点を解消した改良手法を提案する．

文献 [5] の手法における 1 つ目の問題点は，特定のエッジを延長し続けて修復することで他のエッジの修復に悪影響を及ぼす点である．この問題が顕著に表れた例として，標準映像 Caltrain 第 32 フレームの原画像と欠損画像，修復画像を Fig. 1，欠損領域周辺における修復過程を Fig. 2 に示す．ここで，欠損画像における黒色の長方形領域が欠損領域である．Fig. 1(c) より，カレンダーの左端のエッジは，方向を保存して修復されていることがわかる．一方，Fig. 2(b) から (g) より，欠損領域の下部中央付近にある山のエッジが，様々な方向に延長され続けて修復されることで，方向が保存されず，かつ他のエッジの修復に悪影響を及ぼしていることがわかる．

(a) 原画像

(b) 欠損画像

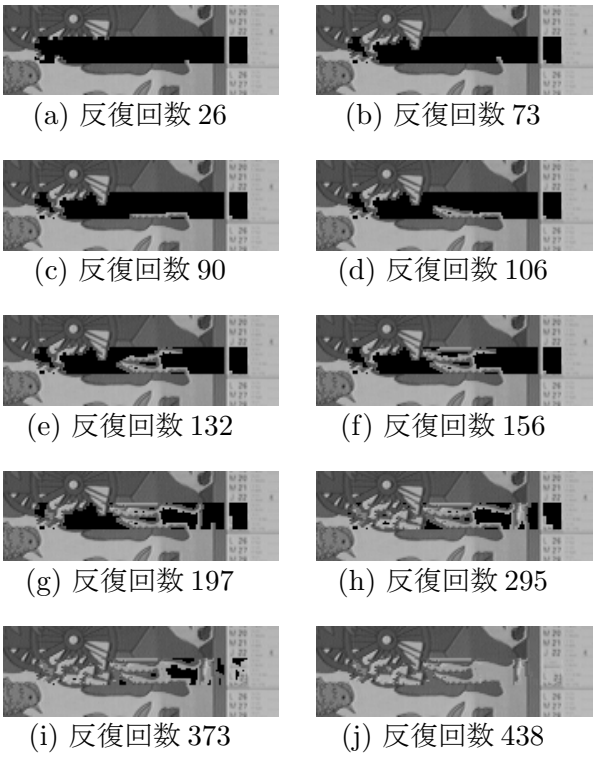
(c) 修復画像

Fig. 1　標準映像 Caltrain 第 32 フレーム

この問題の原因は 2 つある．1 つ目の原因は，次式の Data Term の算出においてパッチ内の輝度値の分散を重みとして掛けていることである．

$$D(p) = \max\left(1, \sum_{q\in(\Psi_p\cap\Phi_\varepsilon)} 1\right) \cdot \frac{\mathrm{var}(\Psi_p)}{|\Psi_p|} \quad (1)$$

ここで，$D(p)$ はピクセル p の Data Term，Φ_ε



(a) 反復回数 26　(b) 反復回数 73
(c) 反復回数 90　(d) 反復回数 106
(e) 反復回数 132　(f) 反復回数 156
(g) 反復回数 197　(h) 反復回数 295
(i) 反復回数 373　(j) 反復回数 438

Fig. 2　修復過程

はエッジマップ*におけるエッジ領域，$\mathrm{var}(\Psi_p)$ はパッチ Ψ_p における輝度値の分散，$|\Psi_p|$ はパッチ Ψ_p のサイズである．パッチは 3×3 ピクセルのため $|\Psi_p| = 9$ となる．なお，エッジの総数は最低 1 を保証している．これより，分散が大きいエッジの Data Term は常に高い値となる．

2 つ目の原因は，Eq. (1) と次式のように，3×3 ピクセル という小さい領域で Confidence Term と Data Term を算出していることである．

$$C(p) = \sum_{q \in (\Psi_p \cap \Phi)} \frac{1}{|\Psi_p|},\ p \in \Omega \text{かつ} \Psi_p \cap \Phi \neq \phi \quad (2)$$

$C(p)$ はピクセル p の Confidence Term，Φ は非欠損領域，Ω は欠損領域である．これより，Confidence Term はいずれのパッチも同等の値となり，非欠損ピクセルが周囲に多く存在するピクセルとそうでないピクセルとの間で差がつかない．この問題を解消するために，提案する手法では，Data Term をエッジであるピクセルの個数のみで定義し，さらに Confidence Term と Data Term の算出時の参照領域を 7×7 ピクセル に拡張した．したがって，Eq. (1) と Eq. (2) はそれぞれ次式のように変更した．

$$D'(p) = \max\left(1, \sum_{q \in (\Theta_p \cap \Phi_\varepsilon)} 1\right) \quad (3)$$

$$C'(p) = \sum_{q \in (\Theta_p \cap \Phi)} \frac{1}{|\Psi_p|},\ p \in \Omega \text{かつ} \Psi_p \cap \Phi \neq \phi \quad (4)$$

ここで，Θ_p はピクセル p を中心とする 7×7 ピクセルの正方形領域である．以上の変更により，特定のエッジが延長され続ける問題を解消する．

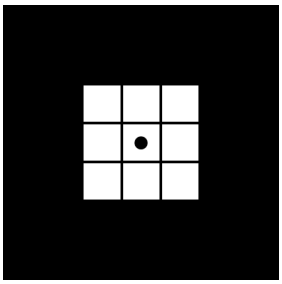
文献 [5] の手法における 2 つ目の問題点は，優先的に修復するエッジの方向を考慮していない点である．自然画像における建造物や植物，人間などの輪郭のエッジは垂直方向に伸びていることが多い．また，データ欠損はスライス単位で発生するため，欠損領域の垂直方向の幅は狭いことが多い．この 2 点より，非欠損領域と欠損領域の境界に対して直角に伸びるエッジは，他のエッジよりも，欠損領域内部においても方向を保存して伸びる傾向があるといえる．以上の理由より，提案する手法では，Data Term の算出における参照領域を，対角線の長さが 7 ピクセルのダイヤ型領域に変更した．したがって，Eq. (3) は次式のように変更した．

$$D''(p) = \max\left(1, \sum_{q \in (\Lambda_p \cap \Phi_\varepsilon)} 1\right) \quad (5)$$

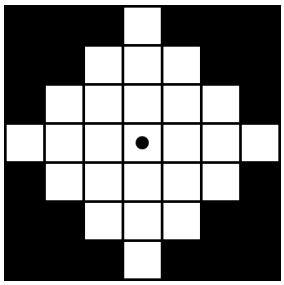
ここで，文献 [5] の手法と提案する手法における参照領域を Fig. 3 の白色領域に示す．ここで，Fig. 3 のブロックはピクセルを表しており，黒色の点で示したピクセルが参照領域の中心である．さらに，エッジマップにおいて，境界に対して平行に伸びるエッジを除外した．以上の改良により，境界に対して直角に伸びるエッジを優先的に修復する．

文献 [5] の手法における 3 つ目の問題点は，エッジマップ作成に要する計算時間が欠損領域のピクセル数に比例するため，修復する画像に

*欠損画像においてエッジであるピクセルを示すマスク



(a) 文献 [5] の手法 Ψ_p



(b) 提案する手法 Λ_p

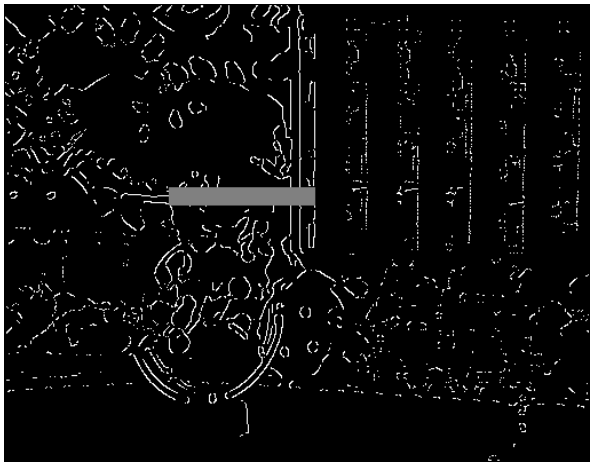
Fig. 3　Data Term 計算時の参照領域

Table 1　エッジマップ作成に要する計算時間

ピクセル数	文献 [6] の手法	文献 [7] の手法
8964 (166×54)	1.45 sec	0.18 sec
204800 (512×400)	40.16 sec	0.35 sec

よって計算時間が異なる点である．この問題の原因は，文献 [5] の手法で用いている文献 [6] のエッジマップ作成手法において，各ピクセルの輝度値を適応的に収束させる処理が必要なためである．この問題を解消するために，提案する手法では，新たに文献 [7] のエッジマップ作成手法を用いた．文献 [6] の手法と文献 [7] の手法を用いて作成された，欠損した標準映像 Caltrain 第 32 フレームのエッジマップを Fig. 4 に示す．ここで，欠損領域は灰色領域で示した．また，前述した問題点より，非欠損領域と欠損領域の境界に対して平行に伸びるエッジは除外した．また，標準映像 Caltrain 第 32 フレームに対して，作成領域のピクセル数を変更してエッジマップを作成した際に要した計算時間を Table 1 に示す．Table 1 より，提案する手法において，文献 [7] の手法を用いることで，修復する画像の欠損領域のピクセル数に比例せず，計算時間を削減できたといえる．

(a) 文献 [6] の手法



(b) 文献 [7] の手法

Fig. 4　欠損した標準映像 Caltrain 第 32 フレームのエッジマップ

3.　動きベクトルベースの修復手法

本節では，本稿で提案する動きベクトルベースの修復手法について述べる．本稿では，移動物体が持つ固有の動きベクトルを修復することを目標としている．この目標を達成するため，注目ブロックの 8 近傍に存在する最頻出動きベクトルによる補間法を提案する．

提案する手法では，まず，欠損したブロックの 8 近傍にあるブロックの動きベクトルを参照し，最も個数の多い動きベクトルを記憶する．ここで，8 近傍のブロックの動きベクトルがすべて欠損している場合は保留とする．次に，記憶した動きベクトルで欠損した動きベクトルを補間

Table 2　実験条件

欠損させる標準映像 Caltrain に関する条件	画像サイズ	512×400 ピクセル
	欠損領域サイズ	128×16 ピクセル
	欠損領域のピクセル数	2048 ピクセル
	動き補償予測ブロックサイズ	4×4 ピクセル
	動きベクトル探索法	全探索法
	動きベクトル探索範囲	水平/垂直方向に ±16 ピクセル
輝度値ベースの修復手法に関する条件	パッチサイズ	3×3 ピクセル
	パッチテンプレートサイズ	5×5 ピクセル
	Confidence Term の参照領域	7×7 ピクセル
	ガウシアンフィルタの標準偏差	2
	Data Term の参照領域	対角線の長さが 7 ピクセルのダイヤ型領域
	パッチの探索範囲	水平/垂直方向に ±18 ピクセル
	パッチの探索法	全探索法
計算機に関する条件	CPU	Intel Xeon 3.40GHz ×8
	RAM	64GB
	OS	CentOS 5.10
	実装言語	MATLAB 2013a

する．この 2 段階の処理を反復することにより，欠損したすべてのブロックの動きベクトルを修復する．

4.　人工的に欠損させた動画像を対象とした修復実験

本節では，2 節と 3 節で提案した手法を用いて，輝度値および動きベクトルが欠損した標準映像 Caltrain を修復した実験について述べる．実験条件を Table 2 に示す．

まず，輝度値ベースの修復実験に用いた標準映像 Caltrain 第 32 フレームの原画像と欠損画像，修復画像を Fig. 5，修復過程を Fig. 6 に示す．Fig. 6 より，カレンダーの左端のエッジや壁紙に描かれた山のエッジが優先的に修復され，かつ特定のエッジが延長され続けて修復されることがないことがわかる．また，Fig. 5(c) より，修復結果として，カレンダーの左端のエッジや壁紙に描かれた山のエッジが，それぞれ方向を保存して修復されていることがわかる．したがって，提案した輝度値ベースの修復手法の目標は達成されていることが確認できた．

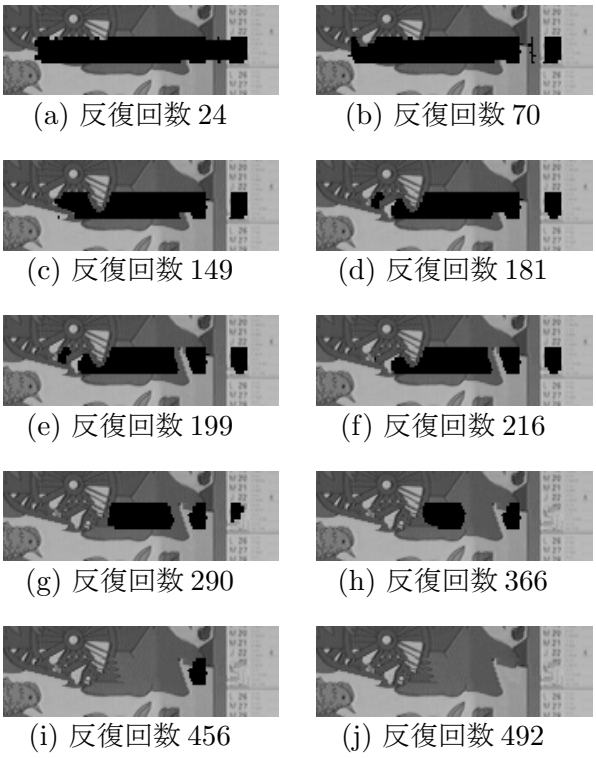
(a) 原画像

(b) 欠損画像

(c) 修復画像

Fig. 5　標準映像 Caltrain 第 32 フレーム

次に，動きベクトルベースの修復実験に用いた標準映像 Caltrain 第 1 フレームの動きベクトルと欠損した動きベクトル，修復した動きベクトルを Fig. 7 に示す．ここで，動きベクトルは



(a) 反復回数 24　(b) 反復回数 70

(c) 反復回数 149　(d) 反復回数 181

(e) 反復回数 199　(f) 反復回数 216

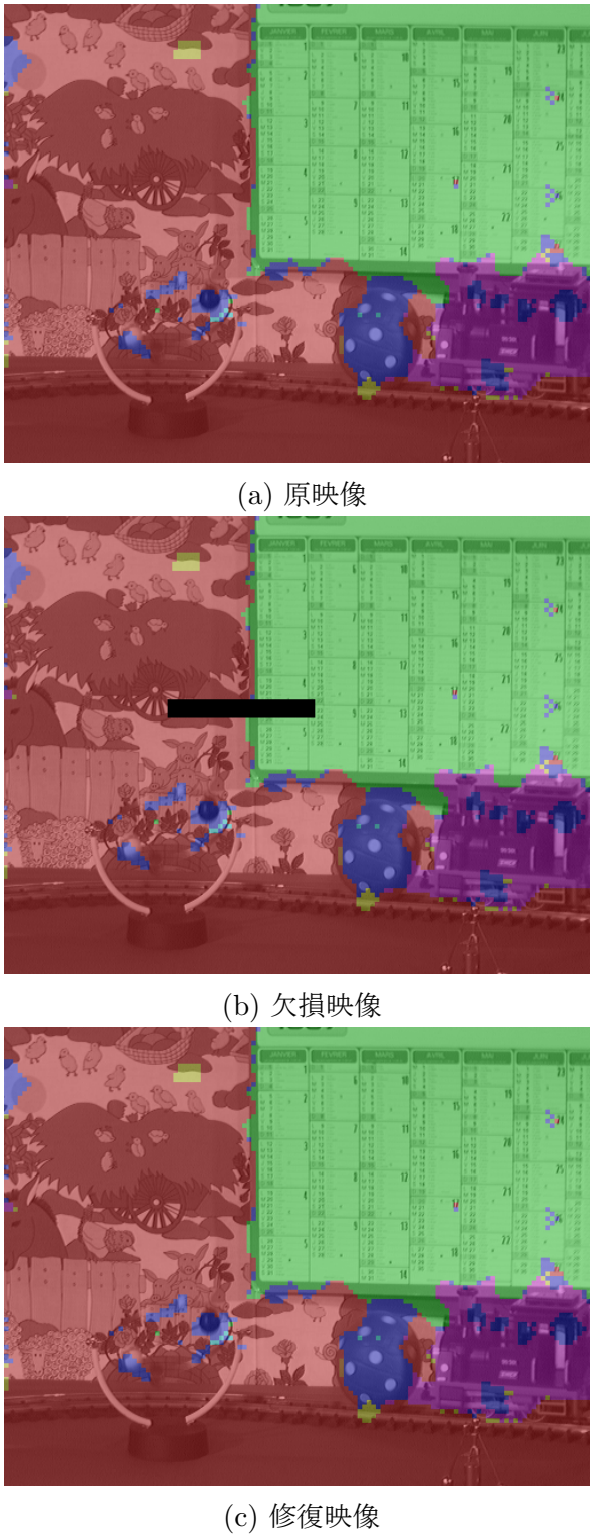
(g) 反復回数 290　(h) 反復回数 366

(i) 反復回数 456　(j) 反復回数 492

Fig. 6　修復過程

動き補償予測ブロックごとに，同一の動きベクトルを色分けで示した．Fig. 7(a) より，壁紙領域に属するブロックの動きベクトルは赤色，カレンダー領域に属するブロックの動きベクトルは緑色で示される動きベクトルを持つことがわかる．Fig. 7(c) において，壁紙領域に属する欠損ブロックの動きベクトルは，壁紙領域の赤色の動きベクトルで修復され，カレンダー領域に属する欠損ブロックの動きベクトルは，カレンダー領域の緑色の動きベクトルで修復されていることがわかる．したがって，提案した動きベクトルベースの修復手法の目標は達成されていることが確認できた．

最後に，計算時間について述べる．本節の修復実験では，輝度値ベースの修復に 18.74 sec，動きベクトルベースの修復に 0.52 sec 要した．ここで，例として，I ピクチャと P ピクチャ，B ピクチャが 1:5:12 の枚数で構成されている DVD-Video 規格を想定する．この比より，B ピクチャの動きベクトルの修復に要する計算時間が P ピクチャの倍と仮定すると，輝度値ベースの修復が必要なフレーム数が 1 の場合，動きベクトルベースの修復が必要なフレーム数は 29 となる．



(a) 原映像

(b) 欠損映像

(c) 修復映像

Fig. 7　標準映像 Caltrain 第 1 フレームの動きベクトル

これより，計算時間をこの比を用いて換算すると，輝度値ベースの修復と動きベクトルベースの修復に要する計算時間は，それぞれ 18.74 sec, 15.08 sec となる．したがって，計算時間の比は 1:0.80 となるので，提案した修復手法では，DVD-Video 規格に基づく映像全体の修復において，輝度値ベースの修復と動きベクトルベースの修復を同等のオーダーで計算できるといえる．

参考文献

[1] M. Bertalmio, V. G.Sapiro and C.Ballester, “Image inpainting,” SIGGRAPH 2000, pp. 417–424, 2000.

[2] A. Criminisi, P. Perez and K. Toyama, “Region filling and object removal by exemplar-based image inpainting,” IEEE Transactions on Image Processing, vol. 13, no. 9, pp. 1200–1212, 2004.

[3] J. Zheng and L. Chau, “A motion vector recovery algorithm for digital video using lagrange interpolation,” IEEE Transactions on Broadcasting, vol. 49, no. 4, pp. 383–389, 2003.

[4] 坂本脩平, 阿部正英, 川又政征, “ミッシングデータが存在する圧縮映像の時空間処理による修復手法の検討,” 情報処理学会第 75 回全国大会講演論文集, no. 1U-2, 2013.

[5] T. K. Shin, N. C. Tang and J. N. Hwang, “Exemplar-based video inpainting without ghost shadow artifacts by maintaining temporal continuity,” IEEE Transactions on Circuits and Systems for Video Technology, vol. 19, no. 3, pp. 347–360, 2009.

[6] D. Comaniciu and P. Meer, “Mean shift: A robust approach toward feature space analysis,” IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 24, no. 5, pp. 603–619, 2002.

[7] J. Canny, “A computational approach to edge detection,” IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 8, no. 6, pp. 679–698, 1986.