社団法人　情報処理学会　研究報告
IPSJ SIG Technical Report

2003－ＡＶＭ－43　（７）
2003／12／18

適応的動きベクトル符号化方式を用いたメモリアクセス低減

鈴木 芳典†

†(株)日立製作所 中央研究所 〒185-8601 東京都国分寺市東恋ヶ窪 1-280
E-mail: †yosinori@crl.hitachi.co.jp

あらまし　動き補償処理に伴うフレームメモリへのアクセスを低減する方式について検討した。最新の動画像国際規格である MPEG-4 AVC | ITU-T H.264 [1]では、6 タップ動き内挿フィルタを用いた 1/4 画素精度動き補償が最小 4×4 サイズの小ブロックで実施される。この高精度の動き補償は、動き予測性能を高める効果は高いが、その一方で動き補償時のメモリアクセスを大幅に増やす要因となる。本報告では、動きベクトルの精度制御と簡易なベクトルスケーリング処理の導入により、符号化処理ならびに復号化処理における演算コストを増加させることなく、また符号化効率を落とすことなく動き補償時のフレームメモリアクセスを低減する方式を提案する。
キーワード　動き補償，メモリアクセス，動きベクトル，MPEG-4 AVC

Reduction of MC Memory Access using Adaptive MV Coding

Yoshinori Suzuki

† Central Research Laboratory, Hitachi, Ltd.　1-280 Higashi-koigakubo, Kokubunji-shi, Tokyo, 185-8601 Japan
E-mail: †yosinori@crl.hitachi.co.jp

Abstract　We propose the new coding method which can reduce the MC memory access using the adaptive MV coding we proposed in [2]. In the latest video coding standard; Advanced Video Coding (AVC) [1], the limitation of no. of MVs per two consecutive MBs is adopted for reducing a memory access relative to the MC process applying both the quarter sample motion interpolation with the 6-tap filter and small MC block-size up to 4×4. However, this limitation may be undesirable for the implementation of AVC codec, since a control of no. of MVs per two consecutive MBs increases an encoder complexity. The propose method can reduce not only the MC memory access but also an encoder complexity.
Keyword　motion compensation, memory access, motion vector , MPEG-4 AVC

1. はじめに

MPEG-4 AVC[1]では、６タップの動き内挿フィルタを用いた 1/4 画素精度の動き補償が、最小 4×4 画素サイズの小ブロック単位で実施される。そのため、１つのマクロブロックの動き補償で必要となる輝度参照フレームへのメモリアクセスは、ワーストケース(16 個の 4×4 ブロックがそれぞれ双方向予測を実施)で 2592 画素((4+5)×(4+5)×16×2)となる。一方、２タップの動き内挿フィルタを用いた 1/2 画素精度の動き補償を採用する MPEG-4 基本 Video 規格[3]では、ワーストケース(４個の 8×8 ブロックがそれぞれ双方向予測を実施)のメモリアクセスが 648 画素((8+1)×(8+1)×4×2)に抑えられている。これは、AVC の新技術がメモリアクセスを４倍にまで増加させることを意味している。

この問題に対処するため、AVC 規格には、動きベクトル数やアクセス画素数を制限する規則が設けられている。しかしながら、状況に応じた符号化制御を必要とするため、この対処法は、符号器の処理コストが増加するという問題を伴う。また、文献[4]に、動き内挿フィルタのタップ数と動きベクトル精度をブロックサイズに応じて適応的に切り替えるという付加符号化制御を必要としない対処法が提案されているが、複数の動き内挿フィルタを導入する必要があり、復号器の実装回路規模を増加させるという問題を伴う。本稿では、適応的動きベクトル符号化方式[2]の応用により、符号化・復号化処理コストを増加させることなく、メモリアクセスを低減する方式を提案する。

以降、２章で AVC 規格と文献[4]の方式、３章で提案方式について説明する。４章にて各方式のワーストケースの比較を行い、５章のシミュレーション実験にて、符号化性能とメモリアクセスコストの観点から提案方式の有効性を検証する。

2. 従来方式

2.1. 動きベクトル数の制限

図１は、AVC 規格[1]の動き補償処理におけるマクロブロック分割モードの種類を示している。基本のマクロブロックモードは４種類あり、8×8 マクロブロックモードについては、更に 8×8 ブロック毎に４種類のサブマクロブロックモードから１種類が選択される。シンタックス上は、4×4 ブロックを 16 個まで持つことができるため、１マクロブロックで符号化される動きベクトルの数は、双方向予測の場合で最大 32 本となる。

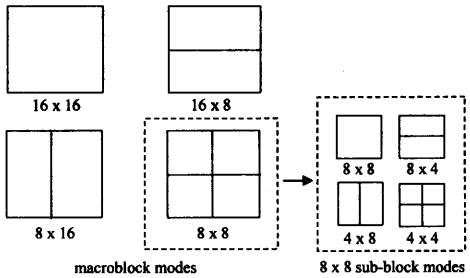

図 1. ブロック分割

規格では、最大ビットレートを 10Mbps 以上とする製品仕様について、「連続する２個のマクロブロックに含まれる動きベクトルの数を 16 本あるいは 32 本に制限する」という規則が設けられている。従って、この規則を適用するエンコーダでは、制限を破らないように動きベクトル数を制御する必要があり、符号化処理が複雑化する。また、このような制限を設けた場合、シンタックス上は正しいが、その制限には違反しているというデータが生成される可能性があり、その取り扱いは難しい。

また、AVC 規格では、低レート向けに「8×8 モードマクロブロック内の 8×8 サブブロックの輝度成分について、アクセス範囲を囲む長方形ブロック(図２の SubMBRect, 斜線部は動き内挿フィルタにより参照されるブロック外画素)を 576 画素以内に制限する」という規則が設けられている。但し、この規則は、メモリアクセスの画素数の制限ではないため、その有効性はエンコーダの構成に依存する。

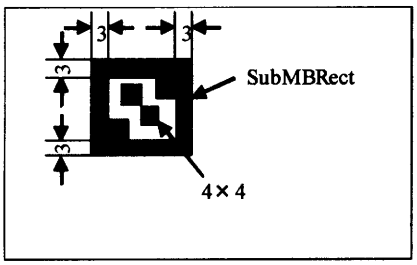

図２.サブ MB のアクセス範囲

本稿は、メモリアクセスの画素数低減を検討の目的としているため、8×8 単位でのメモリアクセスを前提とし、メモリアクセスバンド幅の制限を目的とする SubMBRect に関する議論は行わないこととする。

2.2. タップ数の制限

文献[4]では、AVC 規格を対象として、8×8 マクロブロックモードの 8×4, 4×8 ならびに 4×4 サブブロックについて、「６タップの動き内挿フィルタを用いた 1/4 画素精度の動き補償」の代わりに「２タップの動き内挿フィルタを用いた 1/2 画素精度の動き補償」を適用するという方式を提案している。この方式では、小ブロックほどメモリアクセスが厳しいという点に着目している。AVC 規格の１つのマクロブロック輝度成分におけるメモリアクセスは、１本の実数動きベクトルを持つ 16×16 ブロックの場合には 441 画素(21×21)であるが、16 本の実数動きベクトルを持つ 4×4 ブロックの場合には 1296 画素(9×9×16)まで増加する。そのため、小ブロックのメモリアクセスを低減するというポイントは重要である。しかしながら、文献[4]の方式の実現のためには、６タップと２タップの動き内挿フィルタが両方必要であり、デコーダの実装コストが増加するという問題がある。

3. 提案方式

3.1. 課題

2.1 章に示した動きベクトル数を制限する方法は、ハードウェア実装を中心とし、ワーストケースにおけるメモリアクセスの低減を主要課題とする高ビットレート向けには有効な手段と言える。しかしながら、ミドルウェア実装もターゲットに含まれる低ビットレート向けでは、通常時のメモリアクセス低減も重要な課題と考えられる。特に、AVC 規格が対象とする画素 8 ビット精度の信号では、通常の画素アクセス単位である１ワード（２～４バイト）で２～４画素を同時に扱うことが可能でなるため、水平方向のアクセスワード数よりも垂直方向のアクセスライン数を低減することが、より有効なアプローチと考えられる。

3.2. 対処方法

上記 3.1 章で述べたアプローチを、デコーダ実装コストの抑制という観点も踏まえて考え直すと、「動きベクトルの垂直成分の精度を整数画素に制限する」という方法が、通常時のメモリアクセス低減に適していると判断できる。しかしながら、この対処法を全ての動きベクトルに適用した場合、動き予測性能が著しく劣化するものと考えられる。そこで、本稿では、2.2 章の「小ブロックのメモリアクセスを優先的に低減する」という着目点を導入することで、動き予測性能の劣化を抑制する。また、異なる精度の動きベクトルを、効

率良く、かつ1つの符号化テーブルで符号化するため、適応的動きベクトル符号化方式[2]を利用する。

3.3. 動きベクトル符号化

本章では、まず、文献[2]の適応的動きベクトル符号化方式の概念を説明した後、3.2 章の対処法に適した動きベクトル符号化方法を提案する。適応的動きベクトル符号化方式の本来の目的は、動きベクトルの高精度化と小ブロック分割を同時に適用した場合の問題点を解決することにある。小ブロック分割は、マクロブロック内に複数の動きを持つオブジェクトが含まれる場合に有効となる。しかしながら、1本の動きベクトルの符号量が増加する高精度動きベクトルを適用する方式では、レートと歪の関係から、小ブロック分割モードの選択率が減少すると推測できる。そこで、文献[2]では、動きベクトルの符号化精度をマクロブロック単位で適応的に切り替えることにより、1本の動きベクトルの符号量を調整し、小分割ブロックの選択率を向上させている。具体的には、符号化動きベクトルと予測動きベクトルとの差分動きベクトルの符号化精度をマクロブロック単位で調整し、その精度情報を符号化する。

- 符号化側
  $(MVD_X, MVD_Y) = ((MV_X - P_X) \gg mvda, (MV_Y - P_Y) \gg mvda)$
- 復号化側
  $(MV_X, MV_Y) = ((MVD_X \ll mvda) + P_X, (MVD_Y \ll (mvda) + P_Y)$
  where
  (MV_X, MV_Y); coded MV of integer version,
  (P_X, P_Y); prediction MV of integer version,
  (MVD_X, MVD_Y); differential MV and
  mvda; valiable for indicating accuracy of MVD.

表1. 差分 MV 精度の調整

mvda	差分MV精度
0	1/4
1	1/2
2	1

上記の式にて、MV と P は、実数精度の値を N 倍化して整数値に変換した値である。例えば、1/4 画素精度の動き補償では、各ベクトル成分を4倍した値となっている。表1は、1/4 画素精度動き補償について、差分動きベクトル精度の調整変数 mvda と MVD の精度の関係を示している。この mvda の値を変動させることにより、差分動きベクトルの精度を調整できる。この方式によれば、MVD の値が mvda の値に応じてスケーリングされるため、複数の精度をもつ MV を1個の符号化テーブルで効率良く符号化できる。

このように文献[2]では、差分動きベクトルを対象に動きベクトルスケーリングによる精度調整を行っているため、符号化動きベクトルの精度調整は、予測動きベクトルの精度に依存する。しかしながら、本稿では、差分動きベクトルではなく、動きベクトルの垂直成分を整数精度(小ブロック)と実数精度(その他のブロック)とに場合分けすることが目的となる。そこで、提案方式では、予測動きベクトルにスケーリング処理を適用することにより、予測動きベクトルの精度をブロックサイズに規定される精度に修正する。

符号化側

小ブロックの場合

$$(MVD_X, MVD_Y) = (MV_X - P_X, (MV_Y \gg mvda) - (P_Y \gg mvda))$$

その他のブロック

$$(MVD_X, MVD_Y) = (MV_X - P_X, MV_Y - P_Y),$$

復号化側

小ブロックの場合

$$(MV_X, MV_Y) = (MVD_X + P_X, (MVD_Y + (P_Y \gg mvda)) \ll (mvda)$$

その他のブロック

$$(MV_X, MV_Y) = (MVD_X + P_X, MVD_Y + P_Y)$$

where

(MV_X, MV_Y); coded MV of integer version,

(P_X, P_Y); prediction MV of integer version,

(MVD_X, MVD_Y); differential MV and

mvda is the value that indicates integer accuracy.

このスケーリング方式の適用により、動きベクトルの符号化方法を変えることなく、無駄な符号量を削減し、メモリアクセス低減に伴う性能劣化を抑制する。なお、1/4 画素精度の動き補償を適用する AVC 規格では、mvda の値は2となる。また、本稿では、2.2 章と同様に 8×8 モードの 8×4, 4×8 ならびに 4×4 ブロックを小ブロックとして考えることにする。

この方式では、小ブロックの垂直 MV が整数画素精度に制限されるため、動き予測の実数画素探索に伴う符号化処理量が削減される。また、復号化でも動き内挿に伴う処理量の低減が期待できる。なお、動きベクトルスケーリングに伴う付加処理量は小さいため、処理量の増減には影響ないものと考えられる。

4. メモリアクセス比較

本章では、以下の方式を対象に、1マクロブロック(色差成分も含む)のワーストケースのメモリアクセスコストについて検討する。

- MV Limit: 2.1 章の連続する2個のマクロブロックに含まれる動きベクトルの数を 16 本に制限する方式
- Tap Limit: 2.2 章の小ブロックの動き補償において2タップの動き内挿フィルタを用いる方式
- Prop.: 3 章の提案方式
- No Limit: メモリアクセス制限を行わない方式

各方式のワーストケースは以下のとおり。ワーストケースの選定は、文献[5]に記載されているメモリアクセスのバンド幅推定モデルを利用した。なお、フィルタ境界オーバヘッドは、動き内挿フィルタにより参照されるブロックの外側の画素数を意味する。

- マクロブロックモード(サブマクロブロックモード)
  MV Limit: 8 個の 8×4 双方向予測ブロック
  Tap Limit, Prop., No Limit:
  16 個の 4×4 双方向予測ブロック
- フィルタ境界オーバヘッド
  輝度成分
  - MV Limit, No Limit: 水平 5 画素、垂直 5 画素
  - Tap Limit: 水平 1 画素、垂直 1 画素
  - Prop.: 水平 5 画素、垂直 1 画素
  色差成分
  - 全方式: 水平 1 画素、垂直 1 画素

表 2 は双方向予測を伴う B-picture、表 3 は片方向予測のみを適用可能な P-picture におけるメモリアクセスコスト計算結果を示している。表中の削減率は、No Limit 方式に対する各方式のメモリアクセスワード数削減率を表しており、結果は 3 種類データインターフェース(1 ワードで扱える画素数)についてそれぞれ示している。

表 2 と 3 の結果から、B-picture のワーストケースでは、データインターフェースによる違いはあるものの、3 方式が同程度のメモリアクセス低減効果を有することが分かる。また、P-picture については、MV Limit 方式と比較して、提案方式と Tap Limit 方式の低減効果が大きいことが分かる。これは、提案方式と Tap Limit 方式がアルゴリズムレベルのアプローチであるのに対して、MV Limit 方式は機能の利用制限であることが主な理由と考えられる。つまり、厳しい条件下での効果が高いが、制限のかからない通常時では、大きな低減効果は期待できないと推測される。

表 2．メモリアクセスコスト (B-picture:全体)
(上段：メモリアクセス画素数、下段：削減率)

No. of pixels in a word	No Limit	MV Limit	Tap Limit	Prop.
1	3168	1752	1376	1728
	-	44.7%	56.6	45.5%
2	1824	968	864	1024
	-	46.9%	52.6%	43.9%
4	1152	576	608	672
	-	50.0%	47.2%	41.7%

表 3．メモリアクセスコスト (P-picture:全体)
(上段：メモリアクセス画素数、下段：削減率)

No. of pixels in a word	No Limit	MV Limit	Tap Limit	Prop.
1	1584	1176	688	864
	-	25.8%	56.6%	45.5%
2	912	680	432	512
	-	25.4%	52.6%	43.9%
4	576	432	304	336
	-	25.0%	47.2%	41.7%

5. シミュレーション

本章では、提案方式の符号化性能とメモリアクセス低減効果（通常時）をシミュレーション実験にて検証する。

5.1. 実験条件

表 4 と表 5 に本シミュレーションに使用するテスト画像とテスト条件をそれぞれ示す。これらの条件は、MPEG-4 AVC 規格の標準化作業にて実際に使用されていた。また、表 6 に動きベクトルの値とブロックライン数からフレームメモリへのアクセスライン数を算出する換算ルール、表 7 に動きベクトルの値とブロックサイズからメモリアクセス画素数を算出する換算ルールを示す。本稿では、この表 6 と表 7 を用いて実符号化データの再生値から各 MB の輝度成分のメモリアクセスサイクルを計算し、メモリアクセスコスト低減効果の検証を行う。

表 4．テスト画像

ﾌﾚｰﾑﾚｰﾄ	10fps: IPPP 30fps: IBBP	15fps: IPPP 30fps: IBBP	30fps
画像 (下段は ﾌｫｰﾏｯﾄ と ﾌﾚｰﾑ数)	Container QCIF,300	Paris CIF,300	Mobile CIF 300
	News QCIF,300	Silent Voice QCIF,300	Tempete CIF 260
	Foreman, QCIF,300	-	-

表 5．テスト条件

参照ソフトウェア	JM 5.0
GOP Structure	IPPP... and IBBP...
QP	24,28,32,36
エントロピー符号化	CAVLC
レート・歪最適化制御	ON
参照フレーム数	3
MV 探索範囲	32x32
ダイレクトモード	Spatial

表６．メモリアクセス換算ルール(ライン数)

No. of line	Accuracy of vertical MV component	
	Integer	1/2 or 1/4
16	16	21
8	8	13
4	4	9

表７.メモリアクセス換算ルール(ワード数: 4バイト)

Block size	Accuracy of MV component				
	Horizontal			Vertical	
	Integer (N)		1/2 or 1/4	Integer	1/2 or 1/4
	N%4==0	N%4==1-3			
16×16	4	5	6	16	21
16×8	4	5	6	8	13
8×16	2	3	4	16	21
8×8	2	3	4	8	13
8×4	2	3	4	4	9
4×8	1	2	3	8	13
4×4	1	2	3	4	9

5.2. シミュレーション結果

No Limit 方式、MV Limit 方式ならびに提案方式を対象にシミュレーション実験を行った。表５に示した２種類の GOP 構造と４種類の QP との組み合わせについて、QP 固定の条件で７種類のテスト画像を符号化した。本稿では、MPEG-4 AVC 規格の標準化作業にて適用された客観評価尺度[6]にて符号化性能を評価する。この評価尺度では、２方式のレート歪曲線の性能差を増分 PSNR (DSNR: delta PSNR)とビットレート削減率(BRS: bitrate saving)で評価する。具体的には、比較対象の２方式を、それぞれ４個の QP で符号化し、得られた４個の平均 PSNR と符号化ビットレートの組み合わせから生成される２本のレート歪曲線を比較評価する。表８に IPPP の GOP 構造の結果、表９に IBBP の GOP 構造の結果を示す。表中の Prop.の列は No Limit 方式のレート歪曲線に対する提案方式のレート歪曲線の利得、MVLim の列は No Limit 方式のレート歪曲線に対する MV Limit 方式のレート歪曲線の利得を示している。また、表中の数値は、DSNR のプラスがゲイン、BRS のマイナスがビット量削減を意味している。この結果より、MV Limit 方式、提案方式ともメモリアクセスの制限を行っても、符号化性能はほとんど低下しないことが確認できる。

表８．符号化結果: IPPP

	DSNR [dB]		BRS [%]	
	Prop.	MVLim	Prop.	MVLim
Container	-0.01	0.00	0.23	0.05
Foreman	-0.02	-0.01	0.35	0.26
News	-0.04	0.00	0.63	-0.06
Silent	0.01	0.00	-0.25	0.01
Paris	-0.04	-0.01	0.66	0.19
Mobile	-0.06	-0.01	1.29	0.12
Tempete	-0.04	0.00	0.94	0.08
Ave.	-0.02	0.00	0.42	0.09

表９．符号化結果: IBBP

	DSNR [dB]		BRS [%]	
	Prop.	MVLim	Prop.	MVLim
Container	-0.01	-0.01	0.09	0.11
Foreman	-0.02	-0.01	0.46	0.32
News	-0.04	-0.02	0.60	0.39
Silent	0.00	-0.01	-0.01	0.31
Paris	-0.03	-0.03	0.63	0.58
Mobile	-0.04	-0.01	0.94	0.23
Tempete	-0.06	-0.01	1.43	0.16
Ave.	-0.02	-0.01	0.48	0.30

次に、生成した符号化データと表６、表７の換算ルールを用いて、提案方式の通常時におけるメモリアクセス低減効果を検証する。比較のため、No Limit 方式のメモリアクセスライン数(メモリアクセスワード数)に対する MV Limit 方式と提案方式のメモリアクセスライン数（メモリアクセスワード数）の削減率を計算した。表 10 は QP が 24 の時のフレームメモリへのアクセスライン数の削減率、表 11 は QP が 24 の時のフレームメモリへのメモリアクセスワード（４バイト）数の削減率を示している。ここで、QP が 24 の場合の結果を示したのは、QP が大きくなるほど、小フロックの選択率が減ることを考慮したためである。この結果から、提案方式の通常時メモリアクセス低減効果は、MV Limit 方式に比較して非常に大きいことが分かる。また、IPPP 構造のデータでのメモリアクセス低減効果が IBBP 構造の場合よりも大きくなっていることが分かる。これは、即ち、メモリアクセスの厳しい小ブロックの選択率が、B-picture よりも P-Picture で大きくなる傾向があることを意味している。このことから、４章での推測通り、MV Limit 方式は B-picture で発生するワーストケースではメモリアクセスを低減する効果は高いが、通常のケースのメモリアクセス低減にはほとんど寄与しないことが確認できた。

表 10. メモリアクセスライン数削減率(QP:24)

	IPPP [%]		IBBP [%]	
	Prop.	MVLim	Prop.	MVLim
Container	1.49	-0.44	0.57	-0.14
Foreman	14.83	0.11	7.97	0.50
News	11.81	1.01	5.07	0.85
Silent	10.93	0.99	6.00	0.71
Paris	12.91	1.25	6.81	1.24
Mobile	17.14	1.34	10.11	1.96
Tempete	18.96	1.29	12.43	2.06
Ave.	12.58	0.79	6.99	1.03

表 11. メモリアクセスワード(4 bytes)数削減率(QP:24)

	IPPP [%]		IBBP [%]	
	Prop.	MVLim	Prop.	MVLim
Container	0.18	-0.25	0.30	-0.16
Foreman	11.49	0.22	4.93	0.34
News	6.32	0.41	2.63	0.22
Silent	7.62	0.76	3.98	0.42
Paris	7.98	0.65	3.80	0.50
Mobile	9.94	0.65	5.47	0.93
Tempete	10.96	0.66	6.74	0.98
Ave.	7.79	0.44	3.98	0.46

6. まとめ

垂直方向のメモリアクセスが水平方向よりも厳しいという視点から、小ブロックの動きベクトルの垂直成分を整数精度に制限し、メモリアクセスを低減する方式を提案した。検討結果ならびに実験結果から、提案方式は、ワーストケースのみならず、通常時においてもフレームメモリへのアクセスコストを削減できることが確認できた。また、提案方式は、符号化性能を劣化させないことも確認された。これは、適応動きベクトル符号化方式のスケーリング処理の導入による符号量削減効果に起因する。このスケーリング処理は、低コストで実現できるため、提案方式は、メモリアクセス低減処理に伴うコーデックの実装コスト増加を抑制する効果もある。

文　献

[1] "Text of Joint Video Specification (ITU-T Rec. H.264 | ISO/IEC 14496-10 Advanced Video Coding)," May 2003.

[2] 鈴木, "適応的動きベクトル符号化",信学ソ大, A-6-2, P116, Sept. 2002.

[3] "Text of ISO/ICE 14496-2 Third Edition", March 2003.

[4] 山田, 関口, 浅井, "適応ブロックサイズ動き補償の符号化効率の検討", 2002 画像符号化シンポジウム, pp. 61-62, Nov, 2002.

[5] Adrian Wise and et al., "Model for estimating prediction bandwidth for H.26L", JVT-E093, Oct. 2002.

[6] G. Bjontegaard, "Calculation of average PSNR differences between RD-curves", Doc. VCEG-M33, ITU-T Q6/16, Austin, TX, USA, 2-4 April 2001.