

ソフトウェアベース実時間動画像符号化のための 動き探索処理の一検討

木全英明 八島由幸 小林直樹

NTT サイバースペース研究所

従来、ソフトウェアで実行する動画像符号化の研究において、符号化処理自体の効果的な簡略化による、符号化処理自体の高速化について報告されている。しかし、ソフトウェアの実行環境は個々によって異なるため、実際の符号化処理速度は実行環境に依存して異なる。そのため、実時間性を必要とする符号化において、実時間を保証できない問題が発生する。そこで、実行環境によらずに、実時間で符号化を可能とするため、実行環境に合わせて符号化処理時間を変更できるような仕組みが必要である。本稿では、動き探索時間を目標時間以内にする制御方法を提案する。実験結果により、提案方式は実時間の制限を行わずに符号化した場合に近い画質を実現できることを示す。

A Study on Motion Estimation for Software-Based Real-Time Video Coding

Hideaki Kimata, Yoshiyuki Yashima, and Naoki Kobayasi

NTT Cyber Space Laboratories

In study of Software-based video coding, there are several reports about faster video coding method achieved by efficiently omitting video coding process itself. However, the actual speed of such a software-based video coding depends on implementation environments. Accordingly, under some environments of lower processing speed, the processor can not encode video data in real-time, even if real-time coding is necessary. Therefore the implementation of changing adaptively video coding time dependently on processing speed is expected. In this paper, we propose a new control scheme of motion estimation, which can be controlled adaptively for the target time set in advance. From computer simulation, it is proved that the proposed method achieves nearly the same quality as in case of no limitation of coding time.

§ 1. はじめに

コンピュータの処理速度の高速化により、ソフトウェアで動画像を符号化することが可能となりつつある。ソフトウェアには、実行プログラムをネットワークを介して配布する等の、実行環境の提供の容易さや、実行環境のアップグレードの容易さといった利点がある。今後、このような利点を生かし、ソフトウェアによる動画像の符号化が、より広範囲に行われると予想される。従来、動画像符号化の研究においても、ソフトウェア構成技術の向上による符号化処理の高速化[1]や、符号化処理自体の効果的な簡略化による符号化処理自体の高速化[2][3][4]について報告されている。

しかし、ソフトウェアの実行環境は個々によって異なるため、実際の符号化処理速度は実行環境に依存して異なる。そのため、例えば映像中継などの、実時間で符号化を前提としたアプリケーションでは、符号化を実行するコンピュータの処理速度によっては、実時間を保証できない問題が発生する。そこで、実行環境によらずに、実時間で符号化を可能とする技術が求められる。すなわち、従来のような単に処理の高速化を行うだけではなく、実行環境に合わせて符号化処理時間を変更できるような仕組みが必要である。

実行環境に合わせて、動画像を実時間で符号化するためには、各フレームの符号化時間を、ある設定した時間に収める必要がある。本稿では、その前検討として、特に動き探索方法を制御する方法を提案する。提案方式では、動き探索にSEA(Successive Elimination Algorithm)方法[5][6][7]をベースにして用い、目標時間以内に動き探索が終了する方法について検討した。実験結果により、提案方式は実時間の制限を行わずに符号化した場合に近い画質を実現できることを示す。

§ 2. SEA方式の概要

ここでは、提案方式の動き探索処理でベースとなるSEA方法について簡単に述べる。

2. 1. ブロックマッチング方式

SEA方法はブロックマッチング(BM)方法をベースにしている。BM方法における動き探索では、原画像を $N \times N$ 画素のブロックで分割し、そのブロック内の画素と参照画像のサーチウィンドウ内の画素との間で最も差分の小さい位置を見つける。代表的な差分の評価尺度に絶対値差分和(MAD)がある。参照画像における原画像からの移動位置 (x,y) に対する、MADの計算式を式(1)に示す。ここで、 $f(i,j,t)$ は、原画像(時刻 t)におけるブロック内位置 (i,j) の画

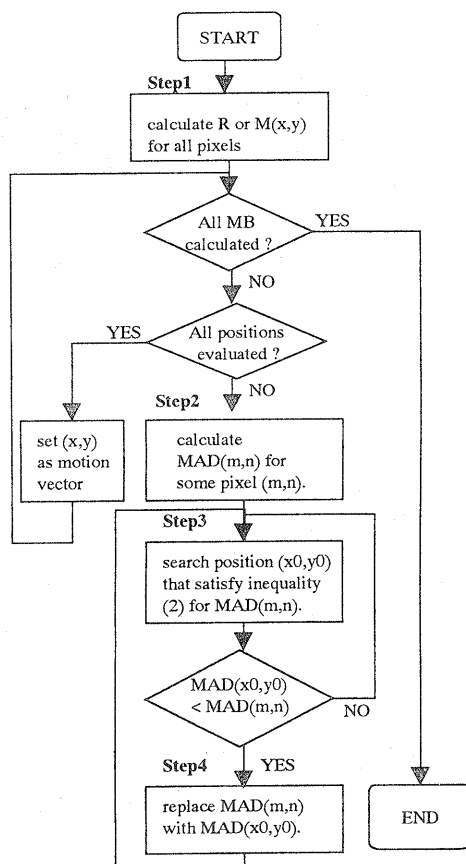


図1. SEA方法の動作フロー

素の輝度値を示し、 $f(i,j,t-1)$ は、参照画像(時刻 $t-1$)におけるサーチウィンドウ内位置 (i,j) の画素の輝度値を示す。 (x,y) をある所定の範囲内で変更して、最もMADが小さい値となる時の (x,y) を動きベクトルとする。

$$MAD(x,y) = \sum_{i=1}^N \sum_{j=1}^N |f(i,j,t) - f(i+x,j+y,t-1)| \quad (1)$$

2. 2. SEA方式による動き探索

SEA方式は、BM方法による動き探索において、数学的な考察を取り入れることによりMADの計算回数を減少できる。このため処理時間の軽減に有効な手法である。具体的には、SEA方式では式(2)を満たす場合にのみMADの計算を行う。 R と $M(x,y)$ はそれぞれ式(3)と(4)により計算される。

SEA方法を使う場合の動き探索の実行方法は例えば次のようになる。フローを図1に示す。

$$|R - M(x, y)| \leq MAD(m, n) \quad (2)$$

$$R = \sum_{i=1}^N \sum_{j=1}^N |f(i, j, t)| \quad (3)$$

$$M(x, y) = \sum_{i=1}^N \sum_{j=1}^N |f(i+x, j+y, t-1)| \quad (4)$$

まず、すべての画素に対してRとM(x,y)を予め求める(ステップ1)。式(2)を満たすかどうかをすべての画素に対して行うので、処理の重複を避けることができる[7]。続いて、探索位置(m,n)を設定して、MAD(m,n)を求める(ステップ2)。このMAD(m,n)に対して式(2)を満たす探索位置(x0,y0)を求める(ステップ3)。MAD(x0,y0) < MAD(m,n)ならば、MAD(m,n)をMAD(x0,y0)で置き換えて(ステップ4)、ステップ3に戻る。そうでない場合には、置き換えずにステップ3に戻る。これを全ての探索位置に対して行う。

§ 3. 提案制御方式

3. 1. 提案方式の概要

動画画像符号化を実時間でを行うためには、各フレームの符号化を与えられた時間以内で行えばよい。図2に1フレームを符号化する時の各符号化処理にかかる時間の割合の例を示す。実験条件を表1に示す。動き探索ではSEA方法による全探索を行い整数位置の動きベクトルを求め、さらにその周辺で半整数位置の動きベクトルを求めた。最初のフレームのみイントラフレーム(Iフレーム)で符号化した。Iフレームでは、AC/DC 予測符号化方法を使っている。IフレームはPフレームのトータル符号化時間に対する符号化時間の割合を示す。PフレームのMakeTableとは、図1におけるステップ1の時間である。MEで示した部分は実際の動き探索処理からMakeTableを除いた時間を示している。(以後この時間をME時間と述べる。)図から、PフレームではME時間が約80%を占めており、このME時間を変更することにより、符号化全体の処理時間を大幅に軽減できることがわかる。すなわち、動き探索処理の処理手順を符号化条件と処理の実行環境に適応して変更すれば、各フレームの符号化時間を所定の時間以内に収めることが可能となる。

動き探索処理には、SEA方法のように全探索処理自体の処理時間を減らす手法の他に、木探索のように、動きベクトル探索確度を犠牲にして動き探索時間を減少する手法もある。提案方式ではSEA方法に一種の木探索を組み合わせて行う。動きベクトル探索確度が低下すると、予測誤差が大きくなるため符号量が多くなる。従って提案方式では、処理時間が小さい程に画質が低下することになる。

提案方式における動き探索処理を次に示す。

表1. 実験条件

画像	Foreman, Stefan (CIF)
符号化方式	MC+DCT
量子化スケール	固定 10 ~ 25
実行CPU	Intel 製 Pentium III 550MHz

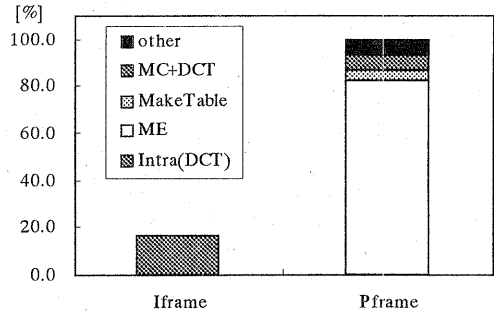


図2. 処理時間の割合

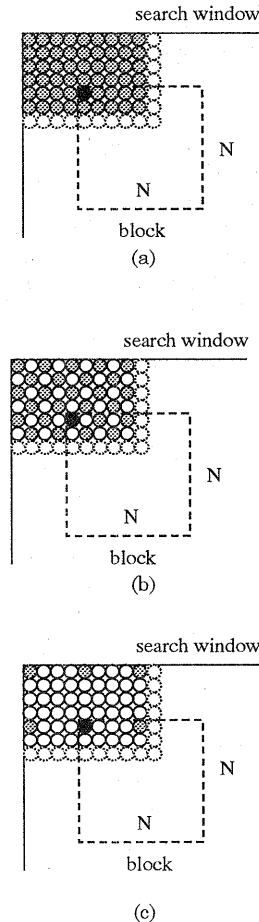


図3. 探索位置パターン

まず、あるマクロブロックに対して整数位置を探索位置の候補としてSEA方法を行い、整数位置の動きベクトルを求める。この時探索位置の候補をマクロブロック毎に変更する。SEA方法の処理高速化を考慮して[7]、図1のステップ1を全画素に対して行った後で、ステップ2からステップ4までの間で、探索位置の候補を変更する。探索位置の候補は、探索範囲の上下左右の対称性を考慮して、図3に示す3パターンにする。丸は参照画像上の画素を示し、そのうち灰色のものは、各探索位置において、原画像ブロック中の左上の画素が対応する画素を示す。破線と黒丸は、それぞれ原画像ブロック位置の例と、そのブロックの左上の画素の例を示す。これらのパターンを変数skipで表すことにして、図3(a),(b),(c)に対応してskip=1,2,4とする。また、探索候補を原画像の同位置のみにする((0,0)ベクトル位置のみ探索する)パターンをskip=0として定義する。

整数位置の動きベクトルを求めた後で、その周囲8個所の半整数位置のMADを求め、最も小さいMADの値となる位置を動きベクトルとする。

3. 2. 実時間符号化処理フロー

提案方式では、与えられた目標時間(target_time)以内に動き探索処理を実行するように、変数skipの値を変更する。実行環境に適した変数skipの初期設定をするために、まず実際の符号化前に、画像内のすべてのマクロブロックを同じskipにした場合のME時間を測定する。以下では測定した時間を、skip=0,1,2,4に対応してtime0,time1,time2,time4で示すことにする。これらの値は、実際の符号化中にも計測していき、画像シーケンス中の、同時刻の画像内容に適した値に更新していく。

マクロブロック毎に、このように各skipにおける平均ME時間を求めながら、次のようを変数skipを決める。変数skipを決定する処理フローを図4に示す。図中のtotal_timeは、現フレーム内において、以前のマクロブロックに費やしたMEの総時間、totalは現フレーム内のマクロブロック総数、countは現フレーム内においてMEを行ったマクロブロック数を指す。

このように、設定した目標時間に対して残された時間内で、残る全てのマクロブロックの動き探索を行うのに、最も探索確度の高いパターンを選択していく。

§ 4. 実験結果と評価

提案方式の効果を確認するため計算機シミュレーションを行った。実験条件は表1と同じである。

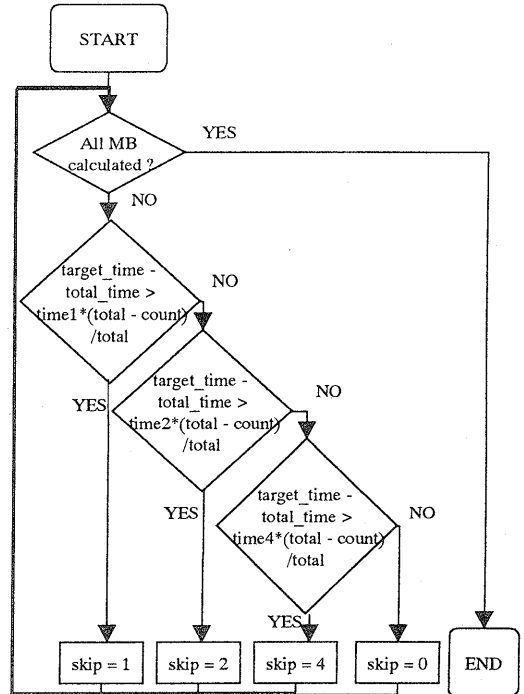


図4. skip 変更フロー

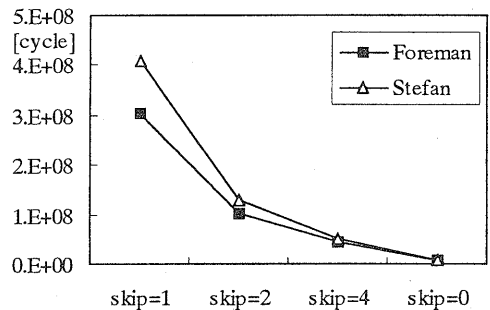


図5. 平均CPUサイクル数

4. 1. 間引き処理による探索時間と画質との関係

まず各探索位置パターンをシーケンスで固定にした場合の実験結果を示す。

各探索位置パターンにおいてMEに費やした、平均CPUサイクル数を図5に示す。なお、このサイクル数を、命令を実行したCPUの動作周波数で除算することにより、実際に費やした時間を求めることができる。図から、変数skipが1,2,4となるにつれて平均サイクル数が、各画像において、同様な比率で減少することがわかる。また、各探索位置パターンにおいて実際にMEにかかる時間は、画像内容に依存して変化することが分かる。

続いて、各探索位置パターンで、画像 Stefan を符号化した場合の、符号量に対する PSNR を図 6 に示す。量子化スケールを固定して実験を行った。変数 skip が 1, 2, 4 となるにつれて、同じ符号量であっても PSNR が低下することがわかる。

4. 2. 指定した時間内での性能

提案方式における変数 skip 決定の処理フローの有効性を検証するため、符号量に対する PSNR を、変数 skip をシーケンスで固定した場合との比較を行った。画像 Stefan で実験した、符号量に対する PSNR を図 7 に示す。図 7 中の

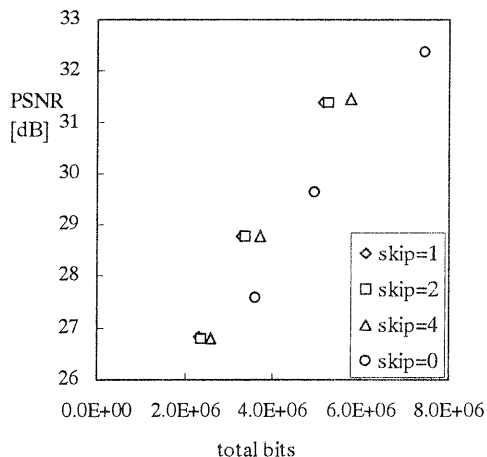


図 6. 符号量に対する PSNR (固定 skip の場合)

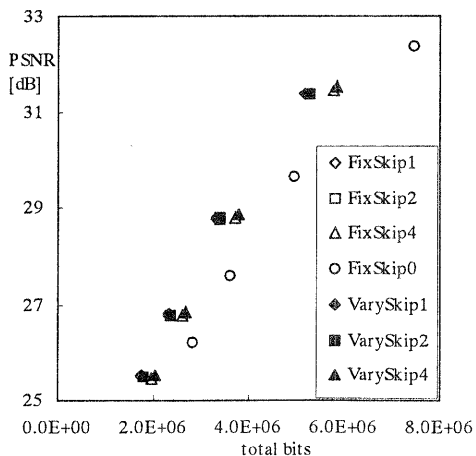


図 7. 符号量に対する PSNR (可変 skip と固定 skip の場合の比較)

FixSkipN (N=0,1,2,4)はそれぞれ、変数 skip をシーケンスで 0,1,2,4 に固定した場合の PSNR を示す。また、VarySkipN (N=1,2,4)は目標 ME 時間を、それぞれ FixSkipN (N=1,2,4)で費やした ME 時間と同じに設定した場合の PSNR を示す。図から、変数 skip を変動する方法は、変数 skip を固定した場合とほぼ同等の PSNR を実現できていることがわかる。すなわち、提案方式は設定した目標 ME 時間以内で、別途実時間の制限を行わずに符号化した場合とほぼ同等の画質を実現できることがわかる。

§ 5. まとめ

本稿では、ソフトウェアによる動画画像符号化において、予め設定した動き探索時間以内で、SEA 方法をベースとして動き探索を行う方法を提案した。実験により、提案方式は実時間の制限を行わずに符号化した場合とほぼ同等の画質を実現できることを確認した。

今後は、探索パターンの選択方法を改良することにより画質を向上する方法について検討を進める予定である。さらに、実際に実行環境に合わせて、各フレームの符号化時間を、ある設定した時間に収め、動画画像を実時間で符号化するシステムについても検討を行う予定である。

§ 参考文献

- [1] J. McVeigh, G. K. Chen, J. Glodstein, A. Gupta, M. Keith, and S. Wood, "A Software-Based Real-Time MPEG-2 Video Encoder," *IEEE Trans. Circuits Syst. Video Technol.*, vol.10, no.7, pp.1178-1184, 2000.
- [2] 大迫史典, 八島由幸, 小寺博, 渡辺裕, 島村和典, "動的演算量スケラブルアルゴリズムによるソフトウェア画像符号化," *信学論(D-II)*, vol.J80-D-II, no.2, pp.444-458, 1997.
- [3] 西和彦, 片山泰男, 正村和由, 持田康典, "計算処理能力に応じた QOS 概念に基づく動画画像符号化処理方式," *信学論(B)*, vol.J82-B, no.1, pp.19-30, 1999.
- [4] B. Erol, F. Kossentini, and H. Alnuweiri, "Efficient Coding and Mapping Algorithm for Software-Only Real-Time Video Coding at Low Bit Rates," *IEEE Trans. Circuits Syst. Video Technol.*, vol.10, no.6, pp.843-856, 2000.
- [5] W. Li and E. Salari, "Successive Elimination Algorithm for Motion Estimation," *IEEE Trans. Image Proc.*, vol.4, no.1, pp.105-107, 1995.
- [6] M. Brünig and W. Niehsen, "A Fast Full Search Matching Algorithm Using Subblocks," *EURASIP EUSIPCO*, vol.2, pp.909-912, 1998.

- [7] 古川潤, 貴家仁志, “MPEG4 のための高速フルサーチブロックマッチングアルゴリズム,” 信学技報, EID98-158, IE98-149, 1999.