2004－AVM－44　(6)
2004／3／4

分散ビデオトランスコーディングにおける VBVバッファの整合性に関する検討

大上 貴充　　渡辺真太郎　　三部 靖夫

株式会社 NTT データ 技術開発本部
〒104-0033 東京都中央区新川 1-21-2 茅場町タワー

あ ら ま し

本稿では，分散ビデオトランスコーディングシステムにおいて，VBV バッファの規格に適合したストリームを生成する方法を検討した．分散トランスコーディングとは，入力の MPEG ビデオを時間軸で分割し，それらを分散処理によってトランスコードしたものを結合して，目的のストリームを生成する方法である．この方法では高速なトランスコードができる一方で，ストリーム生成時の結合の際，VBV バッファがオーバーフロー，または，アンダーフローを起こす可能性があり，ストリームの連続再生を保障できないという問題がある．本稿では，分割された入力の MPEG ビデオが一定区間重複するようにトランスコードする方法を提案し，この問題を回避できることを確認した．

キーワード　　ビデオトランスコーディング, MPEG, VBV バッファ, 符号化レート制御

A study on a compliant VBV Buffer Model for Distributed Video Transcoding

Takamitsu Oogami　　Shintaro Watanabe　　Yasuo Sambe

R&D Headquarters, NTT DATA Corporation
1-21-2 Shinkawa, Chuo-ku, Tokyo, Japan

Abstract

Distributed video transcoding systems divide a video into segments to achieve fast video transcoding by distributed processing. Each segment is transcoded by a distributed PC, and then one video stream is created by concatenating all segments. In this process, the video stream may not be compliant to VBV buffer model. A video stream which is incompliant to VBV buffer model is not guaranteed to playback normally. Thus we propose a new method to create a video stream which is compliant to VBV buffer model by overlapping video-source technique for distributed video transcoding systems. In this paper, we present our method, and experimental results are discussed.

key words　　Video Transcoding, MPEG, VBV Buffer, Rate Control

1　はじめに

多様なブロードバンドアスセス環境や，DVD のような大容量蓄積媒体の普及によって，デジタルコンテンツの流通が活発化しており，一つの映像コンテンツをネットワーク帯域や蓄積媒体の容量にあわせてトランスコードしたいというニーズが高まっている．このようなニーズに応えるために，近年，符号化レートの変換や他の符号化方式へトランスコードする研究が盛んになってきている．例えば，文献 [1][2] では MPEG-2 から MPEG-2 あるいは MPEG-4 へソフトウェア処理でリアルタイムトランスコードする方式の提案がされている．また，文献 [3] では，H.261/H.263 へ高速にトランスコードする方式が提案されている．これらの方式は，動きベクトルや DCT 係数の再利用による処理量の削減や，バッファ遅延を考慮した発生符合量の制御を行うことで高速で高品質なトランスコーディングを実現している．しかし，これらの方式では，仕様の公開されていない符号化方式に対しては，動きベクトルや DCT 係数といった符号化方式に依存する情報を再利用することが困難となるためトランスコードができないという問題がある．

そこで，従来筆者らは，入力のビデオの符号化方式を問わずトランスコードすることと，仕様の公開されていない符号化方式を含めた複数の符号化方式や符号化レートへ同時かつ高速にトランスコードすることを目的として「分散トランスコーディングシステム」を提案した [4]．このシステムは，まず，入力のビデオファイルを時間軸で分割し，それらを多数の PC に転送する．そして，分散処理で復号化と再符号化を行い，一つの PC に回収して結合することで高速なトランスコーディングを実現するものである．しかしながら，このシステムでは，それぞれの PC が独立してトランスコードするため，生成されるストリームが目的の符号化方式におけるデコーダバッファの動作モデルを満たせない可能性がある．その場合，デコーダでの連続再生を保障できないという問題が生じる．

本稿では，MPEG-2 から MPEG-4(CBR) への分散トランスコーディングを取り上げ，以上の問題を解決するため MPEG-4 におけるデコーダバッファ（VBV バッファ）の整合性を保つ方式について検討する．

2　従来方式とその課題

従来筆者らが提案した分散トランスコーディングシステムは，多数の PC を用いた分散処理により高速なビデオのトランスコードを実現した．システム構成は，図 1 に示すように，スケジュール PC，マージ PC，及び複数のトランスコード PC を LAN で接続した形態となる．

このシステムは，まず，利用できるトランスコード

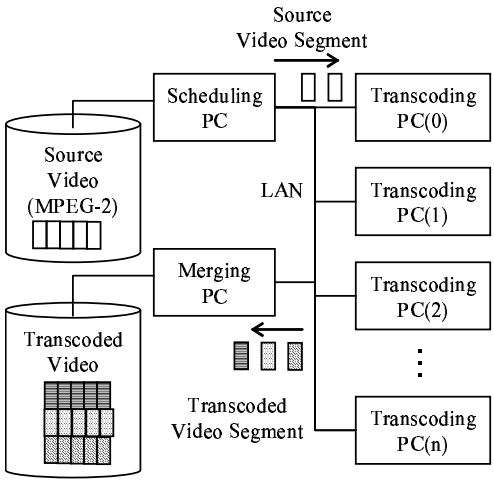

図 1: 分散トランスコードシステムの構成

PC の数に応じて，入力のビデオファイルを時間軸で「分割」し，再生時間の短いビデオファイルをつくる（以降，これをビデオセグメントと呼ぶ）．そして，ビデオセグメントをトランスコード PC に転送する．各トランスコード PC ではビデオセグメントを復号化した後，目的の符号化方式，符号化レートへ再符号化を行う．最後にトランスコードされたビデオセグメントをマージ PC に集めて「結合」し，出力ストリームを生成する．このとき，各ビデオセグメントは独立にトランスコードされているため，それらを結合した際，VBV バッファの整合性が保てない可能性がある．これは，VBV バッファがアンダーフローもしくはオーバーフローを起こし，デコーダでの連続再生を保障できないという問題になる．

この問題に対し，VBV バッファの整合性を保ちつつ，複数の独立した MPEG-2 ファイルを結合する方法が提案されている [5]．この方法は，複数の MPEG-2 ファイルを結合する際，各 MPEG-2 ファイルの先頭フレーム種別（I,P,B ピクチャ）を変更して結合点での発生符号量を VBV バッファの整合性が保てるように調整するものである．しかし，フレーム種別の変更は簡単ではなく，結合処理が複雑になるため，この方法では処理時間を要する．したがって，分散トランスコーディングシステムにこの方法を適用した場合，それが高速処理のボトルネックになる可能性がある．

本稿では，分散トランスコーディングシステムにおいて，高速処理に影響を与えずに VBV バッファの整合性を保つことを課題とし，これを解決するために，ビデオセグメントが一定区間重複するようにトランスコードする方法を提案する．

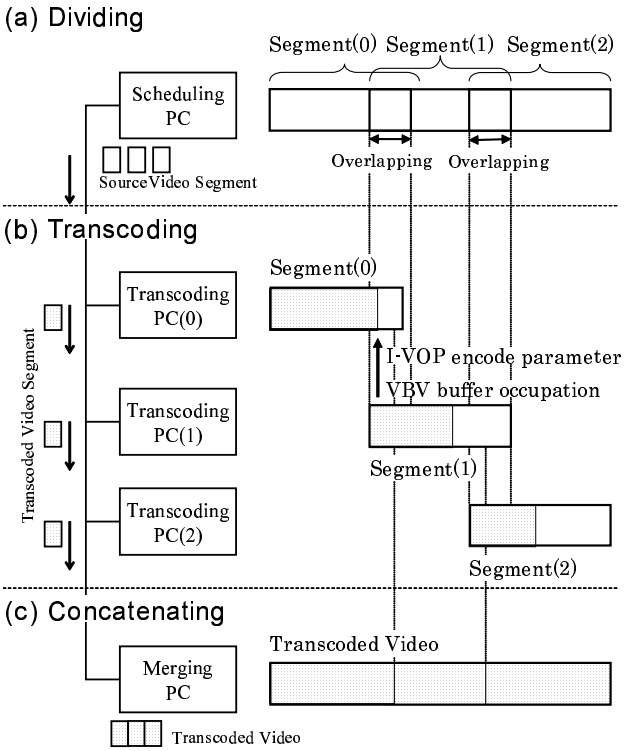

図 2: 重複トランスコーディング方式での処理の流れ

3 提案方式

2 章では，分散トランスコーディングシステムにおいて，VBV バッファの整合性を保ちつつ高速にビデオセグメントを結合することが課題であることを述べた．

提案方式は，ビデオセグメントの一定区間を重複させて分散トランスコードすることで，この課題の解決をはかるものである．なお，各ビデオセグメントの結合の際は，高速性を重視するためフレーム種別を変更するなどの複雑な処理は行わないものとする．その上で，VBV バッファの整合性を保つためには，結合させる 2 つのビデオセグメントにおいて重複区間で VBV バッファ占有量が同じと見なせるポイントを探し，そこで結合処理をすればよい．重複区間のフレームが同じものであるならば，一定区間を経てトランスコードするとその時の発生符号量は，同程度に収束することが期待できる．

この提案方式の具体的な処理フローを図 2 に示す．この図では，入力のビデオファイルを 3 つのビデオセグメントに分割し，3 台の PC でトランスコードする様子を表している．まず，図 2(a) に示すように，ビデオセグメントの終端区間とその次のビデオセグメントの先頭区間が重複するように分割し，各 PC へ転送する．次に，図 2(b) に示すように，各 PC でトランスコーディングを行う．このとき，重複区間では，VBV バッファ占有量を逐次計算し，その遷移の情報を保持しておく．最後に，図 2(c) に示すように，ビデオセグメントの結合処理を行い，出力ストリームを生成する．このとき，重複区間における各々の VBV バッファ占有量の遷移の情報をもとにそれらが同じと見なせるポイントで結合を行う．

以上により，分散トランスコーディングにおける高速処理に影響を与えず，出力ストリームの VBV バッファの整合性を保つことができる．

ここで，本稿で取り上げる MPEG-2 から MPEG-4 へのトランスコーディングを考えると，MPEG-4 のストリームは GOV 構造を持つことができ，さらに双方向フレーム間予測符号化もなされている可能性がある．このときは重複区間で VBV バッファ占有量が同じと見なせるポイントで必ずしも結合処理ができるとは限らない．したがって，そのような状況下でも結合処理ができるようにする工夫が必要となる．以下，その工夫である Open GOV でのストリームの結合方法を 3.1 節で、TM5 改良型のレート制御を 3.2 節で述べる．

3.1 Open GOV のストリーム結合

MPEG-4 の GOV 構造には，GOV 内の符号データのみでデコードが可能な Closed GOV と，GOV 内の符号データだけではデコードできない Open GOV とがある．Closed GOV の場合は，GOV を境界として結合が可能であるが，Open GOV の場合，フレーム間予測が GOV を越えて行われているため，そのままでは結合することができない．しかしながら，一般に Open GOV のほうが符号化効率がよいため，ストリームの先頭以外は，Open GOV 構造とすることが多い．そこで，ここでは Open GOV で結合を可能とする方法を示す．

Open GOV で結合ができない原因は，結合の結果，フレーム間予測で用いる参照フレームのデコード結果が異なってしまうことにある．これは，たとえ重複区間のそれぞれ対応するフレームが同じフレーム種別で符号化されていたとしても，重複区間より前の区間の符号化の影響により，全く同じに符号化されることはないためである．提案の結合方法は，並列処理の利点を生かし、情報を PC 間で共有することで，結合する 2 つのストリームの重複区間に全く同じ I-VOP を作り出し，その I-VOP を基点に結合を行うものである．そのために，I-VOP を参照する B-VOP までをまとめて結合の単位とする．このとき，結合ポイントは GOV の境界ではなく，GOV 初頭の I-VOP を参照している B-VOP の中で最後尾のフレームとなる．

具体例を図 3 を用いて説明する．図 3 は，ビデオセグメントの GOV 構造を示している．ここでは，MPEG の典型的な出力として，N=15，M=3 の構造とした．

図 3 で，GOV の境界であるセグメント (0) の $B_0(72)$ と，セグメント (1) の $I_1(16)$ で結合を行うことを考える．このとき，$B_1(14)$ と $B_1(15)$ の参照フレームは，$P_1(13)$ に変わり $P_0(73)$ となるが，$P_1(13)$ と $P_0(73)$ のデコー

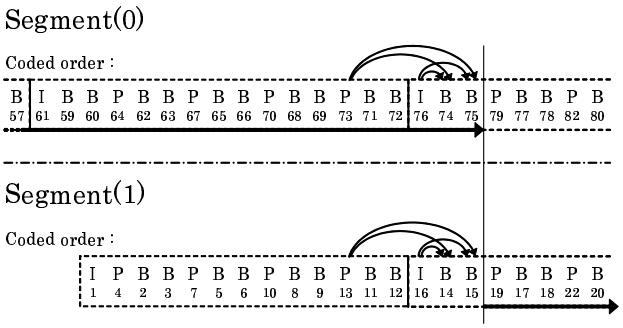

図 3: Open GOV での結合ポイント

ド結果は一致しないため，これらの B-VOP を正常にデコードできる保障はない．

そこで，提案方式では，$I_0(76)$ と $I_1(16)$ のデコード結果を一致させることで，セグメント (0) の $B_0(75)$ と，セグメント (1) の $P_1(19)$ での結合を可能とする．既にトランスコードを終えている $I_1(16)$ の符号化パラメータを用いることで，$I_0(76)$ を $I_1(16)$ と同一の I-VOP に符号化する．

この処理を前節で示した図 2 で説明する．図 2(b) のビデオセグメントの網掛け部分は，トランスコードを終えている部分を示す．ここで，セグメント (0) とセグメント (1) で重複区間をトランスコードするタイミングに注目すると，セグメント (0) の重複区間をトランスコードする時点で，既にセグメント (1) の重複区間のトランスコードは完了していることが分かる．そこで，既にトランスコードを終えているセグメント (1) の I-VOP の符号化パラメータを利用し，セグメント (0) の重複区間での I-VOP のトランスコードを行う．TM5 の場合，I-VOP の目標符号量，仮想バッファの占有量，複雑度を与えることにより，同一の再符号化結果を得ることができる．

以上により，Open GOV での結合が可能となる．

3.2 TM5 改良型レート制御

3.1 節で，Open GOV でも結合が可能となる方法を示した．ここでは，それにより結合を行った場合の結合ポイントで，VBV バッファの整合性を保つように発生符号量を制御する TM5 改良型のレート制御方式を提案する．

提案のレート制御方式は，既にトランスコードを終えている重複区間の発生符号量をもとに，結合ポイントでの VBV バッファ占有量が両ビデオセグメントで同量となるように，目標符号量の調整を行うものである．

提案のレート制御方式の概念図を図 4 に示す．横軸が GOV の番号，縦軸が発生符合量である．この図は，セグメント (0) とセグメント (1) で，3 つの GOV を重複してトランスコードした場合を示している．セグメ

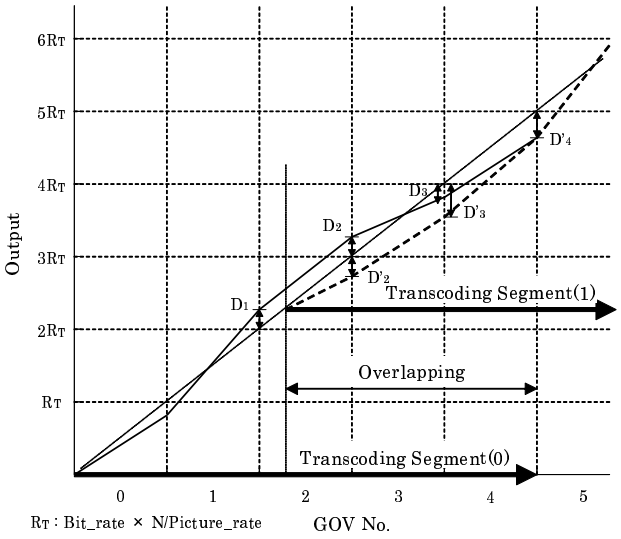

図 4: 提案レート制御の概念図

ント (0) の発生符号量を実線で，セグメント (1) の発生符号量を破線で示す．この実線と破線が交わったところでは，VBV バッファ占有量が同量となり，そのフレームでの結合は VBV バッファに不整合をきたさない．提案のレート制御方式は，既に符号化されているセグメント (1) の発生符号量（破線）にセグメント (0) の発生量（実線）を近づかせる制御である．

以下では，一般的な場合としてセグメント (k) とセグメント $(k+1)$ の重複区間における提案のレート制御方式を式を用いて説明する．$R(n)$ をセグメント (k) における n 番目の GOV の目標符号量，$D(n)$ を次の GOV への繰り越し符号量，$T_b(n,m)$ をセグメント (k) における n 番目の GOV 内で m 番目の B-VOP の目標符号量とすると，それらは次の式で表される．

$$R(n) = B_r \times \frac{F}{F_r} + D(n-1) - D'(n) \quad (1)$$

$$D(n) = R(n) - S(n) + D'(n) \quad (2)$$

$$T_b(n,m) = S'_b(n,m) + \frac{R(n) - S(n)}{M-1} \quad (3)$$

ただし，

B_r ：ストリームのビットレート

F_r ：ストリームのフレームレート

F ：1GOV 内の総フレーム数

$S(n)$ ：1GOV における発生符号量

$D'(n)$ ：セグメント $(k+1)$ における n 番目の GOV での次の GOV への繰り越し符号量

$S'_b(n,m)$ ：セグメント $(k+1)$ における n 番目の GOV で m 番目の B-VOP の発生符号量

M ：P-VOP（GOV の先頭では I-VOP）の間隔

表 1: トランスコーディング条件

Original video	VQEG Test Sequence (NewYork2, Mobile & Calendar, Football, Sailboat)
Frame format	4:2:2, 720 × 486 pixels
Number of frame	260 frames
Frame rate	30 fps
Source video format	MPEG-2 MP@ML CBR
Coding rate	15 Mbps
Frame format	4:2:0, 720 × 480 pixels
GOP structure	M = 3, N = 15
Target video format	MPEG-4 ASP@L3 CBR
Coding rate	768 Kbps
Frame format	4:2:0, 352 × 288 pixels
GOV structure	M = 3, N = 15
VBV buffer size	80 Kbyte

である．

まず，式 (1) について説明する．GOV の目標符号量 $R(n)$ を設定する際，TM5 では前の GOV での目標符号量と発生符号量との誤差を修復するため $D(n-1)$ を加えるが，提案方式ではそこから $D'(n)$ を除いておく．これは，セグメント (k) の GOV 終了時での目標符号量と発生符号量との誤差をセグメント $(k+1)$ での誤差と同じにするためである．しかし，このままでは式 (1) で除いた $D'(n)$ の分だけセグメント (k) の全体の符号量が減ってしまう．そこで，式 (2) で再び $D'(n)$ を加え，全体の符号量に影響を与えないようにしている．次に，式 (3) について説明する．3.1 節で述べた結合方式では，m 番目の B-VOP の直後が結合ポイントとなるため，GOV 先頭での目標符号量からの誤差を調整する．この調整は $R(n) - S(n)$ を B-VOP の枚数に応じて等分に割り当てることによって行っている．

以上の制御により，結合ポイントでの VBV バッファ占有量の差分を抑えることができる．

4 実験結果

ここでは，提案方式が VBV バッファの整合性を保てるストリームを生成できることを実際の映像を用いた実験により確認することを試みる．

実験は，MPEG-2 から MPEG-4 への分散トランスコードを行うものとし，入力のビデオには，VQEG の標準映像（260 フレーム）を用いた．このとき，標準映像のフレーム番号 0 から 192 までを 1 つのビデオセグメント，フレーム番号 90 から 259 までをそれに重複させるビデオセグメントとして，それら 2 つのビデオセグメントを用いて実験を行った．なお，出力ストリームは MPEG-4 の Open GOV 構成とし，符号化レート制御は TM5 に従うものとする．表 1 に本実験の詳細条件を示す．

表 2: VBV バッファ占有量の差分の最小値

	Minimum difference[byte]	
Input stream	TM5	Proposed
NewYork2	184	88
Mobile & Calendar	1744	264
Football	672	608
Sailboat	11176	4216

まず，提案方式によって，2 つのビデオセグメントの重複区間での VBV バッファ占有量の遷移が同じと見なせるようになることを確認する．入力のビデオを標準映像の Sailboat としたときの実験結果を図 5 に示す．この図からは 2 つのビデオセグメントにおいて重複区間での VBV バッファ占有量の遷移がほぼ同じとなっていることが分かる．この結果と比較するために，符号化レート制御が TM5 である場合の同様の実験結果を図 6 に示す．この図からは 2 つのビデオセグメントにおいて重複区間での VBV バッファ占有量の遷移に違いが生じていることが分かる．これらの結果から 3.2 節で述べた提案方式の TM5 改良型レート制御がうまく作用したことが分かる．

2 つのビデオセグメントを実際に結合させる際には，それらの VBV バッファ占有量の差分が最小となるところを選ぶ必要がある．このとき，その差分はできるだけ小さくなることが望ましい．しかも出力ストリームが MPEG-4 の場合は，3.1 節で述べたように GOV 単位で結合ポイントを選ばざるを得ない．

そこで，2 つのビデオセグメントにおいて重複区間での結合ポイントとなる箇所の VBV バッファ占有量の差分を調べた．入力のビデオを標準画像の NewYork2, Mobile&Calendar, Football, Sailboat としたときの実験結果を図 7 に示す．この結果と比較するために，符号化レート制御が TM5 である場合の同様の実験結果を図 8 に示す．これらの結果から提案方式によって，結合ポイントにおける VBV バッファ占有量の差分は全体的に小さく抑えられていることが分かる．さらに VBV バッファ占有量の差分の最小値を調べた．その結果を表 2 に示す．この表から提案方式は符号化レート制御が TM5 である場合に比べ，VBV バッファ占有量の差分の最小値がいずれの場合も小さくなっていることが分かる．

以上から，MPEG-2 から MPEG-4 への分散トランスコードにおいて，提案方式により VBV バッファの整合性を保てるストリームを安定して生成できる可能性があることが分かった．

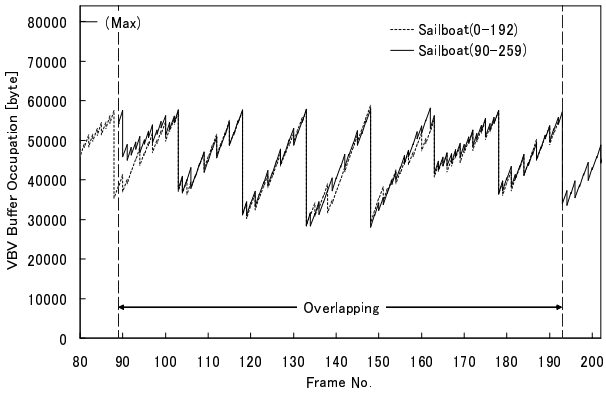

図 5: 提案方式による VBV バッファ占有量の遷移

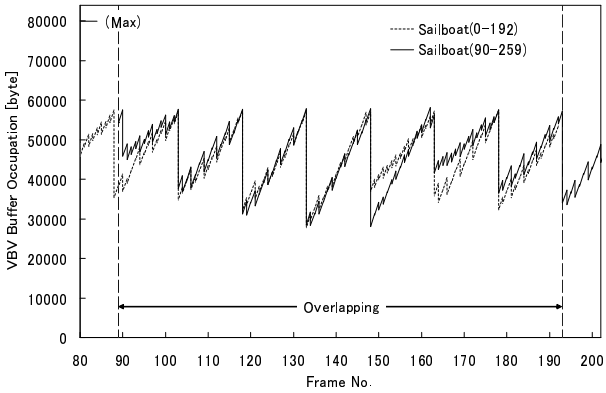

図 6: TM5 による VBV バッファ占有量の遷移

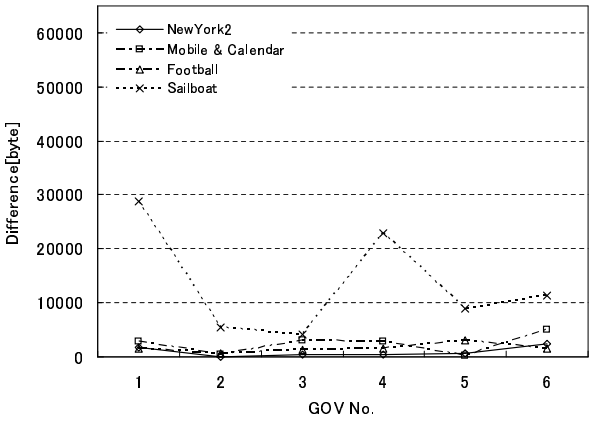

図 7: 提案方式による結合点での VBV バッファ占有量の差分

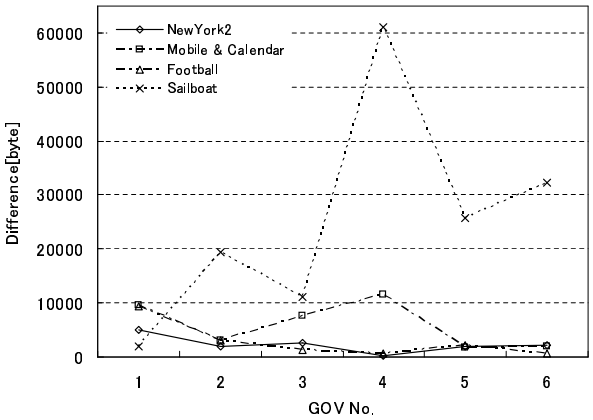

図 8: TM5 による結合点での VBV バッファ占有量の差分

5 おわりに

本稿では，分散トランスコーディングシステムにおいて，VBV バッファの規格に適合したストリームを生成することを目的として，ビデオセグメントの一定区間を重複させてトランスコードする方式を提案した．MPEG-2 から MPEG-4 への分散トランスコーディングを取り上げ，実際の映像を用いた実験により，提案方式が VBV バッファの規格に適合したストリームを安定して生成できる可能性があることを示した．しかしながら，ビデオセグメントの重複区間を長くすると分散トランスコーディングの処理時間に影響を与え，高速処理ができなくなるという問題が生じる．

今後は，ビデオセグメントの重複区間をできるだけ短くしつつ，VBV バッファの規格に適合したストリームを生成する方式の検討が必要である．

参考文献

[1] 笠井裕之, 富永英義, 花村剛, 亀山渉 “低遅延 MPEG-2 ビデオトランスコーダ符号量制御方式”, 信学論 (B), Vol.J83-B, No.2, pp.151-164, Feb.2000.

[2] 阿久津隆史, 木全英明, 清水淳, 八島由幸, 小林直樹 “低ビットレート用 MPEG-2 リアルタイムソフトウェアトランスコーダ「Trampeg」の開発”, 信学論 (D-), Vol.J84-D- , No.6, pp.1084-1093, Jun.2001.

[3] T.Shanableh and M.Ghanbari “HeterogeneousVideo-Transcoding to Lower Spatio-Temporal Resolutions and Different Encoding Formats”, IEEE Trans. on Multimedia, Vol.2, No.2, pp.101-110, Jun.200.

[4] 三部 靖夫, 渡辺真太郎, 于 冬, 中村 太一, 若宮 直紀, “分散処理による高速ビデオトランスコーディングの検討”, 信学技報, IE2002-91, pp.13-18, 2002.

[5] K.Wang and J.W.Woods, “COMPRESSED DOMAIN MPEG-2 VIDEO EDITING”, IEEE, ICME’2000, 2000

[6] ISO/IEC-JTC1/SC29/WG11, “Test Model 5(Draft)”, MPEG93/N0400, 1993.