G-2

映像データに音データを埋め込む動画圧縮技術

Study of video compression technology by embedding the sound data to the image data

○外山紘之[1]，　細野裕行[2]
*Hiroyuki Toyama[1], Hiroyuki Hosono[2]

Abstract: This study is purpose to improve the video compression rate and prevent audio lag. We performs processing for embedding audio data into image data, add this processing to the part of the algorithm of MPEG compression. In this paper, we confirmed that the compression rate of H.261 of 30 seconds can be improved further 4%.

1.　まえがき

デジタルデータの大容量化にともない，圧縮技術は必要不可欠となっている．一般的な動画圧縮方式は，映像データと音データを別々に圧縮し，動画データとしてひとつにまとめたものである．そのため，動画データを再生する際に音ズレが生じる可能性がある[1].

本報告では現在主流の動画コーデックである MPEG の最も基本的な H.261 を用いて画像データに音データを埋め込み圧縮率を高める方法を提案する．これにより，音データ分の容量が削減され，さらに音ズレの低減にも資する．また，本稿では画像を連続再生したものを映像，音が付いている映像を動画とそれぞれ定義する.

2.　利用データ

[映像]

・解像度：1920×1080 pixel
・色：RGB256 段階
・フレームレート：29.97 fps
・再生時間：30 sec
・ファイル形式：H.261

H.261 における主な条件を示す.

・動き予測：片方向予測
・動き補償サイズ：16×16 画素
・動き補償精度：1 画素

[音]

・サンプリング周波数：48,000 Hz
・量子化ビット：24 bit
・チャンネル数：2 ch
・ファイル形式：PCM

サンプリングされた音データの数の単位を個とする.

以上の条件より，フレームレートが約 30 fps，1 秒間の音データが 48000[Hz]×2[ch]=96,000 個であることから，1 フレームの画像に 96000[個]÷30[fps]=3,200 個ずつ音データを埋め込む.

3.　アルゴリズム

H.261 映像に音データを埋め込むアルゴリズムを示す．また，圧縮時のカラーフォーマットは YCrCb の 4:1:1 を用いる．量子化テーブル，ハフマンテーブルは JPEG 規格に則って利用している.

[動画の圧縮]

① 音データを 8 個ずつ離散コサイン変換(DCT)する.
② DCT 係数を全て 127 倍する※1.
③ 小数点以下を四捨五入する.
④ H.261 のエンコードをおこなう[2][3]．このとき，動き補償予測後の交流(AC)係数の色差成分は符号化しない.
⑤ AC 係数の色差成分の上端 1 行と下端 1 行の 1 列目から 100 列目までのブロックにおいて，ビットストリームにする際の，符号化順の 3 番目から 10 番目に DCT 後の 8 個の音データを上書きする．これにより，上端ブロックに 800 個，下端ブロックに 800 個のデータを埋め込むことができる．Cr, Cb 成分それぞれにおいて同処理をおこない計 3200 個のデータを 1 フレームに埋め込む※2.
⑥ AC 係数の色差成分も符号化する.
⑦ ⑤~⑥の処理を全フレームの画像でおこなう.

Fig.1 に上記の圧縮の処理手順を示す.

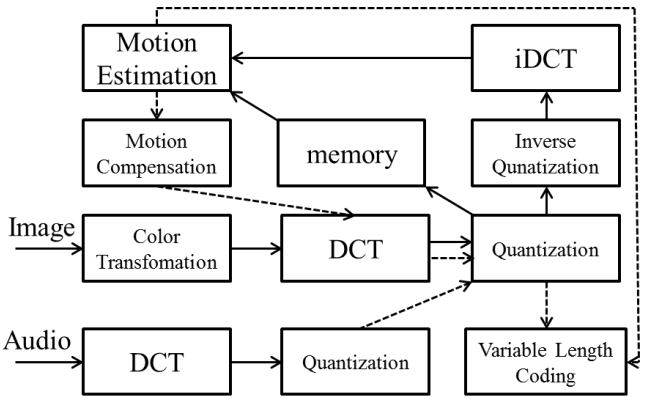

Fig.1 Processing procedure

1：日大理工・院(前)・情報　2：日大理工・教員・情報

※1 : ハフマンテーブルの AC 係数の絶対値の最大値は 1024 である．それに対し，音の DCT 係数の最大値は 8 であることから，音の DCT 係数が AC 係数の最大値をこえず量子化するために 127 倍する．
※2 : 映像再生した際，劣化に気づきにくい位置がよいと考え，音データは上端と下端に埋め込む．

以上のアルゴリズムにより，映像と音をひとつに合成することで，元々の音の容量分を削減できると考えられる．この処理には圧縮率を高める以外にも利点がある．従来の動画方式では，動画再生時に映像データと音データの時間を同期させて再生するものである．しかし，本アルゴリズムでは画像データに音データが埋まっていることから，既に同期している状態であるといえる．これにより，音ズレに強くなる．また，基本は H.261 コーデックを利用しているため，大幅にプログラムの変更をする必要がない．

4.　結果

まず, 画像に埋め込むために前処理①~③をおこなった音データについて述べる．ここで，処理前のビット精度は 24 bit であったが，前処理によりビット精度が 11 bit に下がる．

Table.1 に処理後の圧縮結果の容量を示す．本報告では，これまでの MotionJPEG における処理結果も合わせて載せ，H.261 と MotionJPEG それぞれについて，従来手法(映像データと音データ別々に処理)と本手法を比較する[4]．従来手法の容量は映像データと音データの容量の和である．

Table.1 Compression result

Method of procsssing	Capacity[Byte]	
	MotionJPEG	H.261
Previous method	131,377,819	126,076,750
This method	125,669,645	120,902,006

Table.1 を見ると，どちらも本手法のほうが圧縮後の容量が小さくなっていることがわかる．この結果より，音データ分の容量が削減されているか検討する．

(1)式に本手法の圧縮の向上率を，(2)式に動画データに含まれる音データの割合の計算式を示す．

$$\text{圧縮の向上率} = 1 - \frac{\text{本手法の圧縮率}}{\text{従来手法の圧縮率}} \quad (1)$$

$$\text{音データの割合} = \frac{\text{音データの容量}}{\text{従来手法の容量}} \quad (2)$$

ここで音データの容量は 14 bit 精度であるから，
48,000[Hz]×11[bit]×2[ch]×30[sec]÷8=3,960,000[Byte]
となる．この数値はコーデックによらない．

Table.2 に圧縮の向上率と音データの割合を示す．

Table.2 Data of study

	MotionJPEG	H.261
Ratio of improvement[%]	4.3	4.1
Ratio of audio[%]	3.0	3.3

Table.2 の数値をコーデック毎に比較する．どちらのコーデックにおいても圧縮の向上率が音データの割合を上回っていた．

Fig.2 に実験に用いた映像の 1 フレームの画像を示す．

Fig 2. Image of one frame

MPEG はフレーム間の変化が大きいほど圧縮率が悪くなる欠点がある．しかし，今回実験に用いた映像は Fig.2 のような画像が続くため，場面変化がなく圧縮性能は良い．

5.　むすび

MPEG コーデックを利用した新しい動画圧縮方式を示した．Table.2 の結果より音データ分の容量削減の目的を達成することができた．

今後，動画コーデックを H.262 以降として本手法の処理で対応できるか検討する．また，利用する動画データを場面変化のあるものを用いるなど，サンプル動画のパターンを増やし実験をおこなう．そして，デコード後の再生動画の劣化をピーク信号対雑音比を用いて定量的に評価する．

参考文献

[1] 亀山渉・金子格・渡辺裕：「そこが知りたい最新技術 オーディオ・ビデオ圧縮入門」，インプレス R&D, 2007 年
[2] 越智宏・黒田英夫：「JPEG&MPEG 図解でわかる画像圧縮技術」，日本実業出版社，2003 年
[3] 半谷精一郎・杉山賢二：「JPEG:MPEG 完全理解」，コロナ社，2005 年
[4] 外山紘之・細野裕行：「画像データに音データを埋め込むことによる動画圧縮技術の検討」，日大理工学術講演会，G-7，2013 年