

OMEGAにおけるMPEG-1動画データベースの格納方法

7R-7

野中和明

宇宙開発事業団

増永良文

図書館情報大学

1. はじめに

我々はオブジェクト指向マルチメディアデータベースシステムOMEGAを開発してきた。そこでは、動画や音声といった時間的マルチメディアオブジェクトを時区間としてとらえ統一的なモデル化を行ってきた。一方、動画はMPEG-1により圧縮を施され、GOP単位で格納できる。OMEGAでは動画をオブジェクト化して2次記憶装置に格納して運用するために、どのような格納方法をとるかについてモデルをたて、実装を行い、全体スキーマを明らかにした。本稿ではこの結果を報告する。

2. MPEG-1における動画圧縮スキーマ

一般に動画データは圧縮をかけその記憶容量や伝送量を減らす。OMEGAでは動画の圧縮符号化アルゴリズムとして広く使われているMPEG-1を用いている。

MPEG-1では通常Iピクチャ、Pピクチャ、Bピクチャの3つの圧縮タイプが存在する。Iピクチャとはビデオカメラ等から取り込まれた1フレーム分の動画を一定のアルゴリズムに従って圧縮したものであり、符号化の基盤となるフレームとなる。Pピクチャとは直前のフレームの画像からの変化分だけを取り出したものであるため、データ量は格段に少なくてすむが、Pピクチャ単独での画像の再生は不可能となる。Bピクチャは直前のフレームの画像と直後のフレームの画像の動きの変化分を取り出したものである。

MPEG-1を用いて動画データを圧縮する場合、画素の細かさの設定と圧縮パターンの設定によっても必要なデータ量が変化する。画素の細かさの設定とは、ビデオカメラ等からの動画データを圧縮する際にどの程度鮮明な画像とするかを設定するものである。

圧縮パターンの設定とは、これら圧縮タイプをどのように組み合わせるかを設定することである。たとえば圧縮パターンをIピクチャ1つとPピクチャ2つが並ぶ(IPPP...)のように設定すると、Pピクチャが含まれる分データ量が比較的少なくなるが、3番目のフレームの画像だけを表示させようとすると1フレーム目のデータから順に復号化しなくてはならない。逆に圧縮パターンをIピクチャの連続(IIII...)と設定すれば全てのフレームにおいて単独で表示が可能となるが、データ量は若干増加する。ただし圧縮パターンの設定によるデータ量の増減はほとんどない。

MPEG-1の特長の1つにGOPがある。GOPとは連続した数個から数10個単位でフレームを分割しそれを1つのユニットとして扱うという考えである。これによりフレーム単位での検索より高速な検索が可能となる。

GOPと圧縮パターンはそれぞれ独立して設定を行えるが、1つの圧縮パターン毎にGOPを区切るようにすると効率的な検索、再生が行える。各GOPの先頭のフレームをIピクチャとすればGOP単位で

の再生が可能となるからである。一般にGOPを何フレームで区切るかは自由であり、また圧縮パターンも自由に設定可能であるため、取り込む動画データの種類や性質によって設定を変更できる。

3. クラス階層とクラス定義

2章でのMPEG-1の構造を考慮すると動画クラス階層は次のようになる。まずクラスOMEGATemporalObjectの下に動画であることを表現するための抽象的なクラスとしてクラスVideoが存在する。その下にビデオカメラ等で撮った1本分の内容を表すクラスVideoComponentが存在する。その下に実際に動画データを格納するためのクラスとしてクラスGOPが存在し、高速検索再生を行うために、格納される動画データの1フレーム目をIピクチャにする。また、クラスGOPに格納するフレーム数は固定せずに自由に設定できるようにする。これらの条件も踏まえてクラス定義階層の実装を行った。

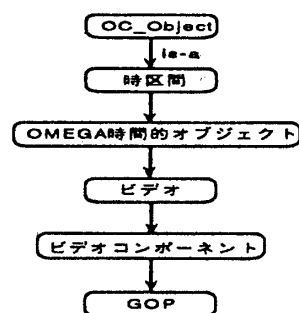


図 OMEGAのMPEG-1動画データベースの格納クラス階層

Figure OMEGA Class Hierarchy for Storing MPEG-1 Video Data

4. GOP当たりのフレーム数によって必要なデータ量の算出

クラスGOPに格納するフレーム数によって必要なデータ量の算出にあたり、次の3つの代替案を考えた。

- (代替案1) 1 GOP当たり1フレームでオブジェクト化する
 - (代替案2) 1 GOP当たり30フレームでオブジェクト化する
 - (代替案3) 1 GOP当たり300フレームでオブジェクト化する
- 1秒間30フレームのレートで撮影される動画300フレーム(即ち10秒分)を格納する場合に必要な記憶容量は以下のようになった。

代替案1	1, 538, 600 bytes
代替案2	1, 503, 800 bytes
代替案3	1, 502, 720 bytes

これらの結果は、生成された各オブジェクトのデータサイズを測ってその合計値を計算することからも予測できる。

以下はONTOS DB 3.1のクラスOC_Entityのメソッドであるsize()を用いて調べた各クラスのオブジェクトに必要なデータサイズである。

OMEGATemporalObject (注1)

= 40 (bytes)

Video (注2)

OMEGATemporalObject + 8

= 48 (bytes)

VideoComponent (注3)

Video + 64

= 112 (bytes)

GOP (注4)

VideoComponent + (videodata + timepointdata + 8)

VideoComponent + (frames * 5000 + frames * 8 + 8)

= 120 + frames * 5008 (bytes)

videodata :

クラスGOPに格納される動画像データの総データ量

timepointdata :

クラスGOPに格納される動画像データの時区間データ

frame :

クラスGOPに格納されるフレーム数

オブジェクト指向モデルでは、サブクラスはスーパークラスのattributeを継承するので、必要なデータサイズは個々で持っているattributeのデータサイズにスーパークラスのattributeのデータサイズも加算される。そのためクラスVideoのオブジェクトのデータサイズはクラスVideo独自のデータサイズの他にクラスOMEGATemporalObjectのデータサイズも加算される。同様にクラスVideoComponentのオブジェクトは、クラスVideoComponent独自のデータサイズに加えてスーパークラスであるクラスVideo独自のデータサイズとクラスOMEGATemporalObjectのデータサイズが加算されているので、結局3つのクラスのデータサイズの合計値がクラスVideoComponentのデータサイズとなる。

動画像をOMEGAに録画する場合は、クラスOMEGATemporalObject、クラスVideo、クラスVideoComponentでそれぞれ1つづつオブジェクトを生成するが、クラスGOPのオブジェクトの生成される数は、録画時にかかったフレーム数をGOPに格納するフレーム数で割ったものとなる。これらを踏まえて録画を行った場合に必要合計記憶容量totalsize (bytes)を計算式にすると以下ようになる。

$$\begin{aligned} \text{totalsize} &= \text{OMEGATemporalSize} + \text{VideoSize} \\ &\quad + \text{VideoComponentSize} + \text{GOPSize} \\ &= 40 + 48 + 112 + (120 + x * 5008) * y / x \\ &= 5008 * y + 200 + 120 * y / x \end{aligned}$$

ここでxはクラスGOPに格納されるフレーム数、
yは録画合計フレーム数。

この式と実際の測定結果とは全く一致するので、録画合計フレーム数とGOP格納フレーム数とによって変化する合計記憶容量はこの式を用いて予測することができる。そこで2時間(7,200秒*30フレーム/秒=216,000フレーム)の動画像を取り込む際に必要合計記憶容量が、上記の代替案の条件によってどの程度変動するのかを予測した結果を以下に示す。

代替案1	約1.11Gbytes
代替案2	約1.08Gbytes
代替案3	約1.08Gbytes

この結果からGOP当たりのフレーム数をいくらしようと合計記憶容量はたいして変わらないことが分かる。

ただしONTOS DB 3.1ではオブジェクトをデータベースに格納する場合、そのオブジェクトを仮想記憶上に展開する必要があるので、オブジェクトの大きさはスワップ領域の大きさの約90%までのサイズが限界となる。例えばスワップ領域を150MBに取ってある我々の現在の環境下ではサイズが135MBを越えるオブジェクトを生成することができない。クラスGOPに必要なサイズは、

$$120 + \text{格納フレーム数} * 5008 \quad (\text{bytes})$$

なので、格納できるフレーム数は26,956フレーム(約898秒)までとなる。

5. まとめ

本稿では、OMEGAにおいて動画像をオブジェクト化して2次記憶に格納して運用するための方法としてMPEG-1の構造を用いたモデルの設計と実装を行い、また格納に必要な記憶容量の実測と予測を報告した。動画像データは他のメディアデータを遙に越える記憶容量の確保が必要となるため、符号化アルゴリズムMPEG-1を用いることによりデータの圧縮を実現した。また高速検索が可能なGOPの機能をOMEGAにも持たせるべくモデルを設計し、クラス定義階層を実装した。このクラス定義では、1GOP当たりに格納するフレーム数も可変させることができ柔軟性に富むものとなった。さらに、このようなクラス定義において実際にオブジェクトを生成した場合に必要な記憶容量を、実測と計算式による予測とで測定し、2時間の動画像データでも十分に格納可能であることを示した。

【文献】安田浩(編著):MPEG/マルチメディア符号化の国際標準,丸善,256P.,1994.

(注1) OMEGATemporalObjectのattribute

OC_Reference object1, OC_Reference object2 ...

リレーションシップをもつOMEGATemporalObjectのオブジェクトその1, その2を参照

OC_Reference mobject ... Videoオブジェクトを参照

int omegaType ... このオブジェクトのリレーションシップがどの関係(equalsかmeets)であることを示す

(注2) Videoのattribute

OC_Reference vcomp ... このオブジェクトの内容がはいっているクラスVideoComponentを指し示す

char *medianame ... このオブジェクトの仮の名前

(注3) VideoComponentのattribute

char *videoname ... 録画したビデオ(動画像)のタイトル

char *videotype ... 圧縮形式(MPEG-1かJPEG)

int totalsample ... 録画したビデオ(動画像)の総フレーム数

double addtime ... 他

OC_List *alist ... このビデオの内容が格納されている全てのGOPオブジェクトのリスト

(注4) GOPのattribute

char data[5000 * FRAMENUM] ... GOPに格納するフレーム数分の動画像データが入っている文字配列

double mediatime[FRAMENUM] ... GOPに格納するフレーム数分の時区間データが入っているdouble型変数

int byte ... このGOPオブジェクトに格納されている動画像データの実際のバイト数

int frame ... このGOPオブジェクトに格納されている動画像データの総フレーム数