データベースシステム 101−16
（1995. 1. 27）

圧縮映像データの格納と検索

友田政明 堀内優希 石川佳治 植村俊亮

奈良先端科学技術大学院大学 情報科学研究科

データベース上で動画像情報を扱う場合に、対象となるデータが圧縮映像であることを想定したモデルの研究はほとんど行われていない。本論文では、動画像データベースの格納データとしてMPEG圧縮データを想定して、これを格納、検索するデータ構造のモデルについて論じる。

MPEG圧縮データは、ユーザがその実体を認識できるフレームに対応した物理構造に加え、それとは非同期な *Pack*、*Packet* 単位による複雑な物理的構造をもつ。本論文で提案するMPEGデータモデルは、このような複雑な物理構造をもつデータを格納するためのモデルである。

さらにこのMPEGデータモデルに基づいて、MPEG映像をあたかも非圧縮映像のように取り出し、VTRで提供されている操作等によりそれを視聴することができるような動画像データベースに関する提案を行う。

Storage and Retrieval Methods of Compressed Video Data

Masaaki TOMODA, Masaki HORIUCHI, Yoshiharu ISHIKAWA and Shunsuke UEMURA

Graduate School of Information Science,
Nara Institute of Science and Technology (NAIST)

This paper describes the data structure, storage and retrieval methods for MPEG data and discusses on the system architecture for the motion-picture database system with MPEG data.

MPEG data also has a complex physical structure that consists of *Pack*, *Packet* as its storage unit, in addition to the physical structure that we can recognize as entity of *Frame*. The proposed MPEG data model has a layered structure.

This paper proposes a retreival method for MPEG data, by which we can retrieve compressed storage data as if it were not compressed. It also discusses on the architecture for a MPEG database system where we can handle data with VTR operation-like methods.

1. はじめに

マルチメディア関連の研究開発が盛んに行われているが、現在最も有望視されているサービスは、VOD(Video On Demand) である[SL92][GKSW91]。このサービスの実現に向けて、特に、ビデオソースをディジタル化して蓄積しておくビデオサーバの開発が盛んである。現状では、ビデオサーバは、タイトル検索程度の操作は提供しているものの、データベース機能といえるほどの機能は装備していない。ビデオサーバにデータベース機能を導入すれば、現状よりさらに柔軟な問合せによる多種多彩な検索が可能となる。

しかし、動画像情報は、従来データベースで扱ってきた文字情報とは異なり、オリジナルのソース情報量が膨大であり、通常のデータベース管理システム (DBMS) で管理するのは非常に困難である。したがって、圧縮されたデータを管理する方が望ましい。

そこで本論文では、まず非圧縮映像をDBMS で管理するために今まで用いられていたデータモデル (以降、在来型データモデル) に基づいて映像スキーマを設計し、その問題点を指摘する。次に、現在国際標準化を終えている MPEG1[ISO93] 映像を DBMS で管理するためのデータ構造のモデル (以降、MPEG データモデル) の提案を行い、それに基づき、データの格納、検索を考慮した MPEG データのスキーマ設計を行う。

2. 在来型の映像データベース

現在までに、様々な研究機関や大学などで、映像情報をデータベースで管理するためのデータモデルの研究開発が盛んに行われている。映像という時間軸メディアは、様々な見方で分析すると、それぞれ特有の『構造』を持っていることがわかる。

2.1 映像情報の多層構造性

映像情報を、以下の『意味構造』、『論理構造』、『物理構造』という三つの異なった側面から分析し、それぞれについて考察する。

定義 1 『論理構造』とは、映像表示装置が具体的にその単位を理解でき、人間もそれ自身の存在を認識できる単位を構成要素とする構造である。

映像の論理構造 そもそも映像 (動画像) とは順序をもった静止画の集合であり、例えば NTSC テレビジョンの場合、$\frac{1}{30}$sec のフレーム、あるいは$\frac{1}{60}$sec のフィールドの順序集合である (以降、フィールドに関する議論は省略する)。したがって、その構造を図示すると、たとえば図 1 のようになる。これまでに提案されているモデルでは、いずれも映像は静止画の列であるということを基本としている。フレームをディジタル化して保存する場合、これらはすべて単にビット列として格納され、そのフレーム内の情報のみでそれ自身のフレームを再生できる。

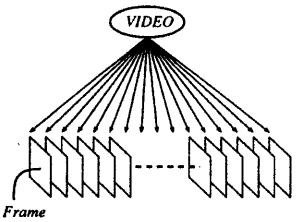

図 1 非圧縮映像の論理 (物理) 構造

定義 2 『物理構造』とは、記憶媒体に実際に格納される単位を構成要素とする構造である。

映像の物理構造 非圧縮映像情報の場合、物理構造は、論理構造とまったく等しい要素から構成されるため、図 1で共用できる。

定義 3 『意味構造』とは、人間が具体的にその意味を理解できる単位を構成要素とする構造である。

映像の意味構造 映像を意味的な側面から分析すると、それは、ある意味を持った *Frame* の順序集合である『場面』の集まりで構成されていることがわかる。『場面』は静止画 *Frame* の順序集合として構成される。ただし『映像』は必ずしも『場面』の順序集合ではなく、定義の仕方によっては『場面』同志が重なったり、入れ子になっていたりする場合もある (図 2)。

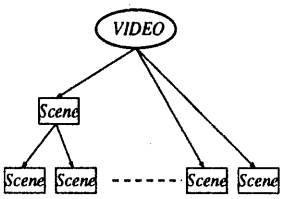

図 2 非圧縮映像の意味構造

ディジタル化された映像情報を、効率良くデータベースで管理するために、現在、図 2のように、複数の連続 *Frame* から成る『場面』(*Scene*) を定義し、各 *Scene* に『場面の意味』を記述する属性を持たせることにより、その属性値を用いて『場面』の検索を行うといった研究が盛んである [OT93][堀内 94]。図 3に『場面の意味』の記述例を示す。

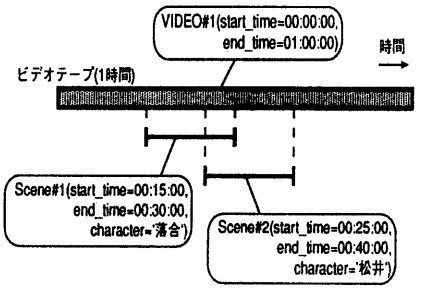

図 3 場面の意味の記述

こうしたモデルはすべて、映像データの意味的、論理的側面までの議論にとどまっており、物理的側面の考察はほとんど行われていない。

2.2 問題点

このような在来型データモデルでデータベースを構築し、ネットワーク経由でデータベースへアクセスを行った場合、ネットワークを流れるデータは非常に膨大な量となる。これはネットワーク資源の有効利用という意味では非効率である。さらに、このようなモデルは、映像が実際に格納されている形式、つまり格納データの物理的な構造をまったく意識しておらず、映像データは 1 フレームずつ独立に格納されており、データアクセス時には、該当フレームのみ抽出すれば、それ自身の情報のみで自己フレームを復元再生可能であるということが大前提となっている。しかし、動画像の情報量は膨大であり、特に長時間にわたる映像、例えば映画ソフトのオンラインデータベース化においては、G(ギガ)～T(テラ)Byte オーダーの情報量となるため、これらをそのまま DBMS で管理するのは非常に困難である。したがって、データベースにはあらかじめ圧縮されたデータを格納しておき、必要な時にそれを復元再生するという方式をとるほうが有効である。しかし、今まで述べてきたような在来型の映像データモデルでは、圧縮という概念はまったく採り入れられておらず、特にその物理構造をまったく考慮していない。

そこで本論文では、データベース格納データとして、動画像圧縮の国際標準である MPEG が最も適当であると考え、現在標準化済みである MPEG1 を対象としたデータベースに関する検討を行った。

3. MPEG データベース

3.1 MPEG1 の物理構造

ここでは、本研究で研究対象とした動画像圧縮方式国際標準 MPEG1[ISO93] の構造について簡単に説明する。

3.1.1 システムストリーム

MPEG1 *SystemStream* は、多層物理構造を持つ。ISO/IEC 11172-1,-2 で圧縮された *CompressionLayer* の MPEG 映像 (*VideoElementaryStream*)、MPEG 音声 (*AudioElementaryStream*) は、それぞれパケットヘッダを伴い *Packet* 化される (*PacketLayer*)。このとき、

各パケットヘッダには、再生、復号のための時刻管理情報等が記述される。そして、それらの *Packet* は複数集められてパックヘッダを伴い *Pack* 化される (*PackLayer*)。このとき、各パックヘッダには、ビデオとオーディオの同期再生に用いられるデコーダの基準時刻参照値 SCR(System Clock Reference) 等が記述される。そして、それらの *Pack* は複数集められてシステムヘッダを伴い *SystemStream* となる (*SystemLayer*)。このとき、システムヘッダには、システム全体の管理情報 (ビットレート、オーディオチャネル数等) が記述される。

3.1.2 MPEG 映像

MPEG 映像 (*VideoElementaryStream*) は、一枚のフレーム自身のみから符号化を行うフレーム内符号化フレーム (*I-picture*)、直前の *P-/I-picture* から予測符号化を行うフレーム間予測符号化フレーム (*P-picture*)、直前直後の *P-/I-picture* から予測符号化を行う双方向予測符号化フレーム (*B-picture*) というそれぞれ符号化方式の異なる 3 種類のフレームで構成されている。それぞれのフレーム (各 *I,P,B-picture*) をＶＡＵ (Video Access Unit, ビデオ復号単位) と呼ぶ。さらにそれらの *I,P,B-picture* は、複数集められて、一つの *GOP*(Group of pictures) を構成している。

I-picture は、フレーム内符号化を行っているため、それ自身の情報のみで元のフレームを復元できるが、*P,B-picture* は、フレーム間符号化を行っているため、それ自身のみの情報では元のフレーム (意味のある独立したフレーム) を復元することができない。この点が、前述の非圧縮映像とはまったく異なる。

3.1.3 MPEG 音声

MPEG 音声 (*AudioElementaryStream*) にも、MPEG 映像と同様に、フレームという概念が存在する。このフレームという単位は、ＡＡＵ (Audio Access Unit, オーディオ復号単位) という、それぞれ単独で復号できる最小単位である。

本論文では、MPEG 音声を、アクセスユニットという MPEG 映像とリンクされた単なるデータ列として扱う。

3.1.4 映像と音声の同期

デコーダにおいて同期再生を行うため、MPEG 映像と MPEG 音声の各アクセスユニットには、タイムスタンプ情報が付与されており、これらの情報により、映像と音声の同期を実現する。

本論文では、このタイムスタンプをスキーマ上に取り入れることにより、同期データアクセスを実現する。

3.2 在来型データモデルに MPEG データを適用した場合の問題点

MPEG データは、それ固有の複雑な物理階層構造をもつため、在来型の映像データモデルに適用した場合、次のような問題点が発生する。

- *I-picture* はそれ自身の情報だけで再生できるが、*P,B-picture* はそれ自身の情報だけでは再生できない。つまり、任意のフレームへのアクセスができない。
- 編集などでフレームを挿抜する場合、フレーム間予測符号化に矛盾が生じる。
- 映像のみに関するモデルでは音声情報を扱えない。また、映像と音声の同期再生も考慮されていない。

以上のような問題点を解決するため、MPEG の物理構造を考慮した MPEG データモデルの考案を行い、スキーマ設計を行った。

3.3 MPEG の多層構造性

MPEG 映像情報を、定義 1,2,3 の側面から分析し、それぞれについて考察する。

MPEG の意味構造 MPEG 映像の場合も、非圧縮映像と同様、これは『場面』に該当する。

MPEG の論理構造 MPEG 映像の場合も、非圧縮映像と同様、これは『*Frame*』に該当する。

MPEG の物理構造 MPEG 映像の場合、これは一つ、あるいは複数の *Frame* に対応する『*I,P,B-picture*』、『*Gop*』と、さらに格納単位である『*Pack*』、『*Packet*』に該当する。

ここではMPEG独特な物理構造に着目し、物理的な2次情報を生成することを目的とする[友田 94]。現在、商用化されているDBMSには、バイナリの長大データを格納するために、*BLOB*(Binary Large Object)クラスが提供されているものもある。ここでは、その*BLOB*クラスのサブクラスとして動画像*MotionPicture*クラス、そのサブクラスとして*MPEG*クラスを定義する。

MPEGのフレーム構造に着目すると、図4のような物理スキーマ(1)を定義できる。

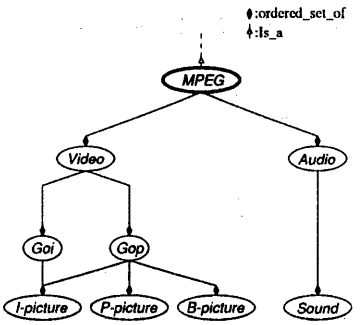

図 4 MPEG物理スキーマ(1)

この物理スキーマ(1)の特徴を次に示す。

(1) *I,P,B-picture*それぞれ1フレームを一つのオブジェクトとする。

(2) *I,P,B-picture*の順序集合を*Gop*オブジェクトとする。

(3) *I-picture*の順序集合を*Goi(Group of I-pictures)*オブジェクトとする。

図4の最下位オブジェクトである*I,P,B-picture*は、図1における*Frame*に一対一対応する。しかし、MPEG1システムストリームは、実際は、MPEGデコーダで再生する際に必要な管理情報を伴った*Pack,Packet*という単位で格納されている。したがって、図5のような物理スキーマ(2)を導入する必要がある。

この物理スキーマ(2)の特徴を次に示す。

(1) 多重化単位である*SystemStream*、*Pack*、*Packet*をそれぞれオブジェクトとする。

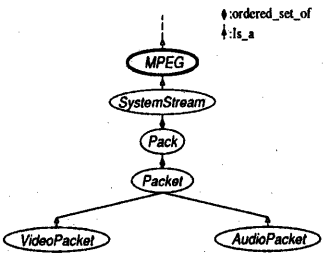

図 5 MPEG物理スキーマ(2)

(2) 映像パケット*Video-packet*と音声パケット*Audio-packet*をオブジェクトとする。

(3) タイムスタンプ情報により、映像パケット*Video-packet*と音声パケット*Audio-packet*を関連づける。

また、図4,5の物理スキーマ(1),(2)は、図6のように統合することができる。

物理スキーマ(1)における映像、音声の各アクセスユニットは、物理スキーマ(2)と密接に関連している。したがって、最下位オブジェクト同志(*I,P,B-picture*と*Video-packet*、*Sound*と*Audio-packet*)をスキーマ上で関連づける。

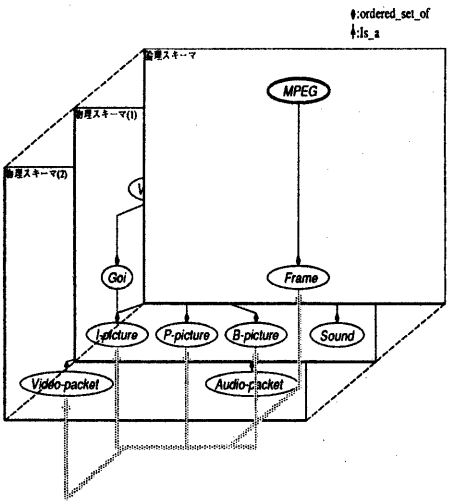

図 6 MPEGスキーマ

3.4 MPEGオブジェクトのクラス定義

図6に基づいたクラス定義の例を次に示す。オブジェクトの時間順序関連を指定するために順序集合 *(ordered_set_of)* 型を導入する。実際、MPEGバイナリデータは、*MPEG* クラスのインスタンスとして格納される。したがって、それ以外のオブジェクトはすべて、*MPEG* オブジェクトへのポインタを持つ抽象オブジェクトである。↓印は属性、メソッドを継承していることを示す。

3.4.1 MPEGクラス

MPEG クラスは、MPEGデータ (バイナリ) を格納するためのクラスである。また、その属性、メソッドをすべて *MotionPicture* クラスから継承する。

```
CLASS MPEG:{
  PhysicalAttribute:
    ↓ component:  ordered_set_of(Frame)
    ↓ start:      Frame
    ↓ total:      INTEGER
  Method:
    ↓ PLAY(start(Frame),end(Frame))
    ↓ STOP
    ↓ PAUSE
    ↓ F-SEARCH
    ↓ B-SEARCH
}
```

3.4.2 フレームクラス

Frame クラスは、論理スキーマの最下位クラスであり、物理スキーマ (1) における最下位クラスを参照する。

```
CLASS Frame:{
  PhysicalAttribute:
    refer_to:     {I,P,B}-picture
  Method:
    DISPLAY
}
```

3.4.3 システムストリームクラス

SystemStream クラスは *Pack* の順序集合で構成される。

```
CLASS SystemStream:{
  HeaderInformation:
    BitRate:      REAL
    VideoCH:      INTEGER
    AudioCH:      INTEGER
  PhysicalAttribute:
    component:    ordered_set_of(Pack)
    start:        Pack
    total:        INTEGER
  Method:
    read_data
    write_data
}
```

3.4.4 パッククラス

Pack クラスでは、デコーダで *Video* と *Audio* を同期再生するために用いられる基準時刻参照値 SCR 属性を定義する。

```
CLASS Pack:{
  HeaderInformation:
    SCR:          HEX
    MuxRate:      REAL
  PhysicalAttribute:
    component:    ordered_set_of(Packet)
    start:        Packet
    total:        INTEGER
  Method:
    ↓ read_data
    ↓ write_data
}
```

3.4.5 パケットクラス

Packet クラスは、*Video-packet* と *Audio-packet* をサブクラスにもつ。

```
CLASS Packet:{
  HeaderInformation:
    PTS:          HEX
    DTS:          HEX
  Method:
    ↓ read_data
    ↓ write_data
}
```

3.4.6 ビデオパケットクラス

Video-packet クラスは、*I,P,B-picture* から参照されている。また、タイムスタンプ情報をもとに、同期再生されるオーディオパケットとリンクしている。

```
CLASS Video-packet:{
  HeaderInformation:
    ↓ PTS:        HEX
    ↓ DTS:        HEX
  PhysicalAttribute:
    is_referred_to  {I,P,B}-picture
    link_with       set(Audio-packet)
  Method:
    ↓ read_data
    ↓ write_data
}
```

3.4.7 ビデオクラス

Video クラスは *Gop*、*Goi* をサブクラスにもち、再生メソッドを持つ。

```
CLASS Video:{
  PhysicalAttribute:
    ↓ component:  ordered_set_of({Gop,Goi})
    ↓ start:      {Gop,Goi}
    ↓ total:      INTEGER
    playback-time: TIME
    total-bit-size: INTEGER
  Method:
    ↓ PLAY
    ↓ STOP
    ↓ PAUSE
    ↓ F-SEARCH
    ↓ B-SEARCH
}
```

3.4.8 GOPクラス

編集 (*Gop* の順序変動) による孤立 *Gop* を明示するため、識別属性 (flag) を定義する。

```
CLASS Gop:{
  PhysicalAttribute:
    ↓ component:  ordered_set_of ({I,P,B}-picture)
    ↓ start:      {I,P,B}-picture
    ↓ total:      INTEGER
    flag:         {closed, broken}
    previous:     Gop
    next:         Gop
  Method:
    ↓ PLAY
    ↓ STOP
    ↓ PAUSE
}
```

3.4.9 GOIクラス

Goi は *I-picture* の順序集合で構成され、*Gop* と同様、再生メソッドを持つ。

```
CLASS Goi:{
  PhysicalAttribute:
    ↓ component:  ordered_set_of( I-picture)
    ↓ start:      I-picture
    ↓ total:      INTEGER
    previous:     Goi
    next:         Goi
  Method:
    ↓ PLAY
    ↓ STOP
    ↓ PAUSE
}
```

3.4.10 Iピクチャクラス

I-picture クラスは、物理スキーマ (1) の最下位クラスであり、物理スキーマ (2) における最下位クラスを参照する。また、論理スキーマにおける最下位クラスから参照される。*I-picture* は自身のみでフレームを復元できるため、再生メソッドを持つ。

```
CLASS I-picture:{
  PhysicalAttribute:
    ↓ previous:   I-picture
    ↓ next:       I-picture
    belong_to:    Gop,Goi
    refer_to:     set(Video-packet)
    is_referred_to: Frame
  Method:
    DISPLAY
}
```

3.4.11 Pピクチャクラス

P-picture クラスは、物理スキーマ (1) の最下位クラスであり、物理スキーマ (2) における最下位クラスを参照する。また、論理スキーマにおける最下位クラスから参照される。*P-picture* は自身のみでフレームを復元不可能であるため、再生メソッドは持たない。

```
CLASS P-picture:{
  PhysicalAttribute:
    ↓ previous:   P-picture
    ↓ next:       P-picture
    belong_to:    Gop
    refer_to:     set(Video-packet)
    is_referred_to: Frame
}
```

3.5 問合せの例

ユーザが遠隔ビデオサーバに問合せを行う場合の例を以下に示す。サーバ側のデータベース (DATABASE) に、野球中継 (1 試合) の MPEG 映像が格納されているとする。

i) ユーザは、ユーザ端末から (疑似) 自然言語等によりデータベースに問合せを行う。

> 松井がホームランを打った場面を見たい

ii) その (疑似) 自然言語は SQL 等のデータベース問合せ言語に変換される。

```
select   OID, 開始時間, 終了時間
from     DATABASE
where    character='松井'
           and
         scene='ホームラン'
```

iii) 生成した SQL をメッセージ化してネットワークへ送出する (ユーザ端末→ビデオサーバ)。

iv) ビデオサーバはこのメッセージを受信し、データベースへアクセスを行って候補となるものすべてを抽出する。

(OID,	開始時間,	終了時間)
(1001,	00:10:15,	00:13:00)
(1015,	01:05:20,	01:07:50)
(1212,	01:44:10,	01:47:05)
⋮	⋮	⋮

v) 抽出した [OID, 開始時間, 終了時間] をネットワークへ送出する (ビデオサーバ→ユーザ端末)。

vi) ユーザは問合せに合致した場面映像の集合を確認し、そのなかから見たいものを選択する。

vii) 選択された映像の OID 番号をネットワークへ送出する (ユーザ端末→ビデオサーバ)。

viii) ビデオサーバはこの OID 番号を受信し、実際に格納されている MPEG オブジェクトへの変換を行う。

ix) ビデオサーバはこのMPEGオブジェクトに関し、データベースにアクセスを行ってMPEGデータを取得する。

x) ビデオサーバは取得したMPEGデータをネットワークに送出する(ビデオサーバ→ユーザ端末)。

xi) ユーザはこのMPEGデータを受信し、端末でデコードを行って映像を視聴する。

ivにおいて、DBMSに問合せを行うことにより、まずデータベースから『松井がホームランを打った』を意味属性値に持つ場面を仮想オブジェクトとして生成し、その[OID, 開始時間, 終了時間]の組の集合を抽出する。

次に、viiiにおいて、[開始時間, 終了時間]それぞれをフレーム番号に対応させ、実際に格納されている各アクセスユニットオブジェクト(*I,P,B-picture*)にいったん変換する。この後、実際に格納されている実オブジェクト(*Pack,Packet*)へ対応づけて、場面の仮想オブジェクトを生成し、データベースへ問合せを行ないMPEGデータを取得する。

このviiiの処理を伴う点が、非圧縮映像を格納した在来型の映像データベースへアクセスする場合と大きく異なる点である。

4. まとめ

MPEG1データをDBMSで管理するためのデータモデルの提案と、そのモデルに基づいたMPEGスキーマの設計を行った。このスキーマ上でMPEGデータを格納、検索することにより、ユーザは、従来通り *MotionPicture* クラスだけを認識しておけば、それ以下の構造はまったく意識する必要がない。つまり、その格納データが特別な構造を持ったMPEGであることをまったく意識せずに、非圧縮映像のときと同様に映像データベースにアクセスすることが可能になる。

5. 今後の課題

今後の課題としては、検索における効率性を考慮した各オブジェクトの具体的な格納法およびOODBへの実装について検討を行ない、さらにMPEG2へと展開する。

参考文献

[GKSW91] A. D. Gelman, H. Kobrinski, L. S. Smoot, and S. B. Weinstein. A store-and-forward architecture for video-on-demand service. In *Proc. IEEE International Conference on Communications 1991(ICC91)*, Vol. 2, pp. 842–846, June 1991.

[ISO93] *ISO/IEC 11172-1 : 1993(E) Information technology - Coding of moving pictures and associated audio for digital storage media at up to about 1.5 Mbit/s - Part 1 : Systems*, 1993.

[OT93] E. Oomoto and K. Tanaka. OVID: Design and Implementation of a Video-Object Database System. *IEEE Trans. on Knowledge and Data Eng.*, Vol. 5, No. 4, pp. 629–643, August 1993.

[SL92] Joe Sutherland and Larry Litteral. Residential Video Service. In *IEEE Communications Magazine*, pp. 36–41, 1992.

[堀内 94] 堀内優希, 友田政明, 植村俊亮. ビデオデータベースにおけるデータの格納、検索手法. 情報処理学会第49回全国大会, 3W-09, September 1994.

[友田 94] 友田政明, 堀内優希, 植村俊亮. 圧縮映像情報の物理データモデル. 情報処理学会第 49 回全国大会, 3W-05, September 1994.