ビデオシーケンスの構造化

滝川 啓[†] 権田 亜紀子[††] 福留 治隆[†††]

ランダムアクセスが可能なビデオ再生システムにおいて，内容に基づいて分割されているビデオ素片を複数個，時系列あるいは同時に組み合わせて再生し，連続したビデオシーケンスを生成する構造化手法を提唱する．各素片の論理的なつながりを示す論理構造記述，再生時の素片の接続・重畳におけるフェード・ワイプなどの付加的な属性の記述と，素片データを各々独立に記述することにより，シナリオや特殊効果の変更など，ビデオシーケンスの編集が容易になる．また，ビデオ・オン・デマンドシステム等において，論理構造や再生属性をテンプレートにして，素片のみを入れ換えたシーケンスを自動生成することができる．本論文では特にシーケンスの構造記述方法について述べ，他の構造記述方法との比較，再生方法についても触れる．

Video Sequence Structuring

KEI TAKIKAWA,[†] AKIKO GONDA[††] and HARUTAKA FUKUTOME[†††]

A video sequence structuring method for a random access video system is proposed. Using the method, a continuous video sequence is produced from number of video elements which are segmented based on context, concatenating and/or mixing them. Describing a logical structure, additional attributes such as fading or wiping effects between successive video elements, and video elements separately, a scenario and presentation effects can easily be modified in video sequence editing. Moreover, in a VoD system, sequences can be automatically produced from selected video segments, using predetermined logical structure and presentation attributes as a template. This paper mainly describes the structure description method. The comparison with other structuring methods and the playback method for this structuring method are also discussed.

1. まえがき

ディジタルビデオ技術を始めとするマルチメディアデータ操作技術，インターネットを始めとするネットワーキング技術の進展により，インタラクティブマルチメディアサービスの技術基盤が整いつつある．しかしながら，現状では情報作成を人手に頼っているため，情報作成のコストが高く，マスメディア的なサービスしか成り立たない．したがって，個々人のニーズに対応しうる真のインタラクティブサービスは困難な状況にある．

情報作成コストを低減させるためには，たとえばデータベース検索の結果を用いてリアルタイムにマルチメディア情報を生成する技術が必要になる．このためには，情報の構造化を行うとともに，シーケンス生成のルール（メタテンプレート）を作成する必要がある．

本論文では，従来構造化の検討があまり進んでいないビデオシーケンスを取りあげ，ビデオ素片を単位とする構造化を試みる．初めに，ビデオシーケンスの内部を並列あるいは直列に接続された複数の素片データの集合として表すとともに，内容依存，表現テクニック依存の情報を分離して構造の自動生成や編集を容易にする記述方法を提案する．その後，時間軸をベースとした構造化に比べて記述データ量が少ないこと，論理構造記述から時間構造記述への変換，さらにはそれに基づくビデオの再生が可能であることについても触れる．

本構造化手法は，デスクトップビデオ編集やVoDにおける自動シーケンス合成などに適用可能である．また，ビデオシーケンスの構造化にとどまらず，二次元空間構造を含むマルチメディアデータ全般，さらには三次元表現にも拡張が可能と考えられる．

† NTTソフトウェア株式会社ニュービジネス事業本部第二事業部
New Business Sector, Sector 2, NTT Software Corp.
†† 株式会社日立製作所知的所有権本部
Intellectual Property Office, Hitachi, Ltd.
††† 株式会社グラフィックス・コミュニケーション・ラボラトリーズ
Graphics Communication Laboratories

2. ビデオ構造の記述

過去，ビデオシーケンス（オーディオも含む．以下，同じ）の記録に何らかの構造を導入しようとする試みがいくつかなされている．アップル社の QuickTime では，Track と呼ばれる複数のオーディオ/ビデオデータ相互の時間関係を実データとは別にファイルヘッダとして記述する．このような形式にすることによって，データ量の多いビデオデータのカット&ペースト等のハンドリングが容易になる利点がある[1]．Avid Technology 社では Open Media Framework（OMF）と称して，同様に階層構造を持つ構造記述部と各種属性値，データ実体を分けて管理する方式を提唱している．花村らは時間軸上のシーン構成と，シーナリ構造と呼ぶフレーム内のオブジェクト構造とを記述するビデオドキュメントアーキテクチャを提案している[2]．Weiss らはビデオの構造をビデオ素片単位とし，それらの関連を Video Algebra と称する演算式により表現する方法を発表している[3]．

また，Hardman ら[4]は論理構造をベースにしたマルチメディアオーサリングが時間軸ベースのオーサリングに比較して編集や再利用に有利であると主張している．

一方，文書処理の分野では ODA[5]や SGML[6]に見られるように論理構造と割付構造の分離が行われ，いわゆるプロセッサブルドキュメントの概念が一般化してきている．まえがきに述べたように，ビデオシーケンスの場合においても「プロセッサブル」の概念が必要と考えられ，文書処理における実施例が参考となる．

筆者らはプロセッサブルなビデオを指向してビデオシーケンスの論理構造に着目した構造化を検討してきた[7]．

3. 論理構造に基づく構造化

監視ビデオやテレビ会議などを除き，人為的に作られるビデオシーケンスは基本的には内容，視点などの違いに基づく構造を持っていると考えられる．ただし，編集後のシーケンスではフェード，ワイプなどによって別個の素片（場面）が連続的につながっているように見えたり，ミキシングやクロマキーによって同時に表示されたりすることがある．そこで，まず基本的な構造（論理構造）と修飾的な構造（再生時間や再生効果などに関する属性：再生属性）に分離して考える．これにより，内容依存の情報（再生時間）や制作者に依存する情緒的情報（再生効果）を切り離すことが可能になる．

3.1 論理構造

本論文で扱う構造は，同一スクリーン上に表示される同一サイズのビデオシーケンスに限定する．論理構造はビデオシーケンスの意味的な構造，たとえば導入部，本題，結びなどの区分，あるいは背景と字幕などの合成に関する構造をツリー構造で表す．また，再生属性に関するデータ，およびデータ実体は論理構造とは別のデータとして扱う．

論理構造は，構成要素（Part，Scene など．総称してオブジェクトと呼ぶ）が1個ずつ時系列で再生される直列再生と，複数オブジェクトが同時に再生される並列再生の組合せで表現される．基本的な論理構造は最上位階層を Package とし，ストーリの組立てに対応して Scene，Cut，Clip に細分するツリー構造である．ツリーの各リーフを時系列（直列）に再生する．これに並列再生のオブジェクトを組み合わせることができるように，補助的な階層として Package の下に Part，Scene の下に Subscene，Cut の下に Subcut を加える．

構造の一般形（メタテンプレート）を**図1**に示す．ここで，四角形は並列再生のオブジェクトであり，同一階層で並列に何個でも複数存在し得るオブジェクトを示す．角丸四角形は直列再生のオブジェクトであり，直列に何個でも複数繰り返しうるオブジェクトを示す．さらに，網掛けで示されたオブジェクトは実データを参照するオブジェクトであり，中抜きのものは構造記述のための仮想的なオブジェクトである．

この一般形によれば，並列に再生されるものどうしの直列や直列に再生される複数のオブジェクトと単一のオブジェクトの並列再生も可能である．これらは特にオーディオとビデオの関係を記述する場合に有効で

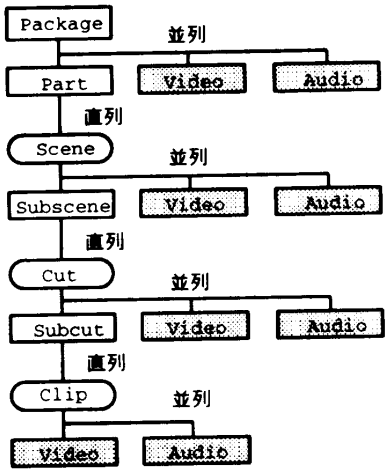

図1 論理構造の一般形
Fig. 1 The general form of the logical structure.

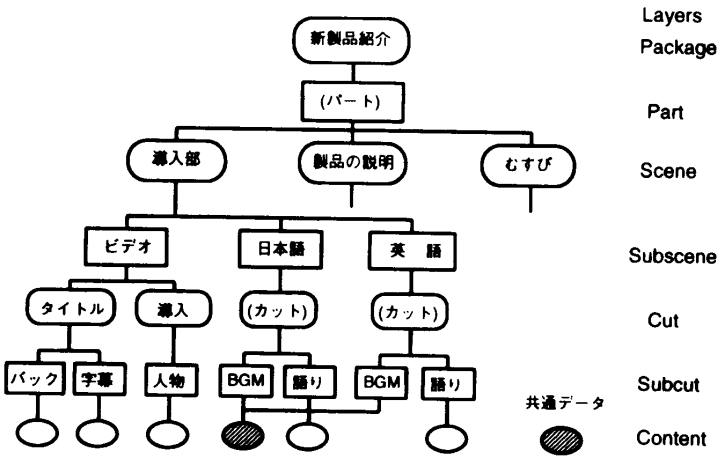

図2 論理構造のサンプル
Fig. 2 A sample logical structure.

ある．図2に具体的な論理構造のサンプルを示す．ここで，(パート)，(カット) は，ストーリー上の意味は持たないが，下位構造を記述するために便宜上挿入したオブジェクトである．また，この例では Clip 層は使用していない．

「日本語」と「英語」は並列に再生されるように指定されているが，再生系によっては，どちらか一方を選択して再生したり，別々に再生することも可能と考えられる．

3.2 再生属性

再生属性は論理構造で表されたオブジェクトおよびオブジェクト間の接続や並列再生における効果を指定する属性である．指定できる属性には以下のようなものがある．なお，具体的記述法については3.5節に述べる．

- 単独のオブジェクトに対する指定
  フェードイン，フェードアウト，スタートオフセット
- オブジェクト間の直列再生に対する指定
  ワイプ，クロスフェード，オーバラップ（ミキシング），スタートオフセット
- オブジェクト間の並列再生に対する指定
  クロマキー，はめ込み（ワイプ），ミキシング，スタートオフセット

これら再生属性データは論理構造データと独立しており，アイデンティファイヤにより間接的に参照される．このようにすることによって論理構造を変更することなく再生効果だけ変更することが容易であるほか，再生属性を共通利用することも可能になる．

なお，単独のオブジェクトに対する属性を Presentation-Style，複数のオブジェクト間の直列・並列再生に関する属性を Linkage-Style と呼ぶことにする．ただし，両者一連の ID を付与し，論理構造からはどちらも presentation-style-id によって参照する．

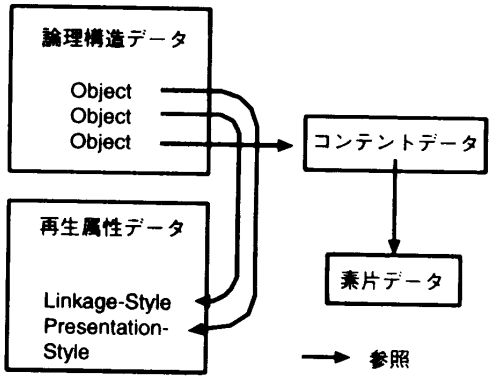

図3 各記述要素の相互関係
Fig. 3 The relationship between structure elements.

3.3 コンテント

コンテントはオーディオ・ビデオデータを記述するものである．構造は MHEG[8] の content-class をベースとしている．コンテントは大別するとプロファイル部と素片データ部に分かれ，プロファイル部には書誌情報や符号化方法に関する情報が記述される．素片データ部はプロファイル部においてさらに外部ファイル参照の形で指定される．

3.4 各記述要素の相互関係

論理構造，再生属性，コンテントの各要素は図3に示すように，各々独立に記述される．再生属性とコンテントは，論理構造の中の各オブジェクトから参照される．再生属性はアイデンティファイヤによって，またコンテントはファイル名によって参照する．このようにすることにより，各々の要素の独立性を高めることができる．たとえば，複数の再生属性を用意しておけば，参照する再生属性のアイデンティファイヤを変えるだけで，種々の特殊効果を選択することができる．

3.5 記 述 法

構造記述は人間による直接記述ではなくGUI等を介して自動生成されることを前提としている．ODA[5]にならい，各オブジェクトに関する記述をシーケンシャルに並べたフラットな構造とした．この方法はツリーをネスト構造で記述する方法に比べて，オブジェクトの記述位置が自由である特徴がある．したがって，オブジェクトの挿入や削除が容易である．図2に示したオブジェクトの一部を記述した例を以下に示す．表記法はASN.1値表記法[9),10)]による．

object { — } が1つのオブジェクトを記述している．object-typeはそのオブジェクトの種類を示し，object-descriptorの中でアイデンティファイヤ，名称，下位オブジェクト（subordinate）を示す．また，再生属性を指定する場合はpresentation-style-idに再生属性のアイデンティファイヤを指定する．

```
object {
  object-type   package,
  object-descriptor {
     object-identifier     0,
     user-visible-name
       "新製品紹介",
     subordinate          {0}
  },
},
object {
  object-type   part,
  object-descriptor {
     object-identifier    0 0,
     user-visible-name
       "(パート)",
     subordinate     {0, 1, 2}
     presentation-style-id    {0}
  },
},
object {
  object-type   scene,
  object-descriptor {
     object-identifier  0 0 0,
     user-visible-name
       "導入部",
     subordinate       {0, 1, 2}
  },
},
object {
  object-type   subscene,
  object-descriptor {
    object-identifier   0 0 0 0,
    user-visible-name
      "ビデオ",
    subordinate        {0, 1},
  },
},
object {
  object-type   cut,
  object-descriptor {
     object-identifier 0 0 0 0 0,
     user-visible-name
       "タイトル",
     subordinate         {0, 1},
  },
},
object {
  object-type   video,
  object-descriptor {
     object-identifier 0 0 0 0 0 0,
     user-visible-name
       "バック",
     object "background.cnt"
  },
}
```

また，再生属性のデータ記述の例を以下に示す．この例は複数のオブジェクト間の接続効果を指定するlinkage-styleを規定しており，style-id 0と名付けられている．この再生属性は，先の記述例中のpart（パート）において，presentation-style-id 0として参照されている．この例ではpart（パート）配下の3個のSceneのうち，始めの2個を遷移時間2秒のクロスフェードでつなぎ，後の2個を遷移時間1秒の水平ワイプで切り替える特殊効果を与える．

linkage-directiveは効果の対象となるオブジェクトをlinkage { - }で指定し，効果の種類をlinkage-typeで指示している．

```
linkage-style {
  style-id   0,
  linkage-directives {
    linkage  {
      {object  subordinate 0},
      {object  subordinate 1}
    },
    linkage-type cross-fade 2000,
    linkage  {
```

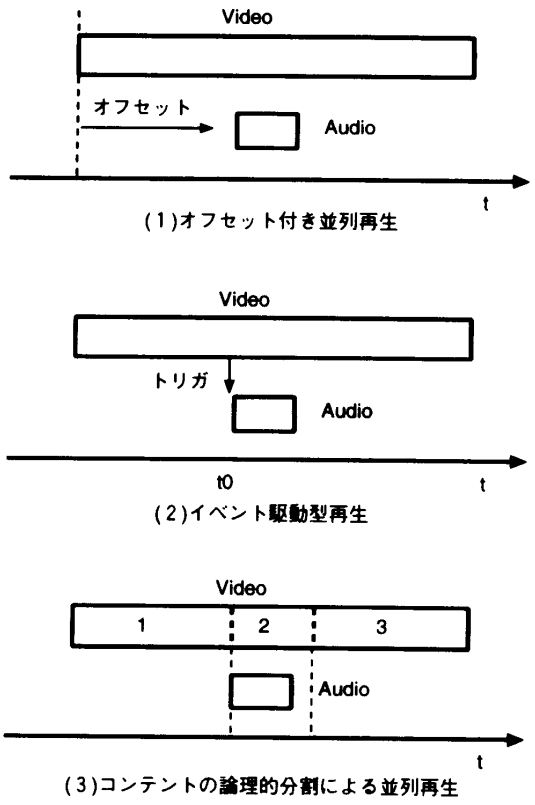

図4 詳細同期方法
Fig. 4 Precise synchronization methods.

```
    {object  subordinate 1},
    {object  subordinate 2}
  },
  linkage-type horizontal-wipe 1000
},
}
```

3.6 コンカチネーション

たとえばビデオに効果音などを付加させる場合，連続したビデオの途中にオーディオ素片を同期させる必要がある．これを解決する方法として，

(1) ビデオの先頭からオーディオの開始点までのオフセット時間を指定する方法
(2) 時間軸上に設定されたマーカに同期させてオーディオ再生を開始させる方法
(3) 連続しているビデオの同期を必要とする部分を構造記述上で分割しておき，オーディオオブジェクトと当該のビデオオブジェクトの並列構造として記述する方法

の3方法が考えられる．図4に各方法を図示する．

(1)は再生属性にオフセット時間を指定することにより実現できるが，絶対時間により同期を記述するため，素材の変更に対応できない．(2)は時刻によるイベントのトリガが必要になるのでツリー構造記述の基本方針と合わず，新たな再生メカニズムも必要になる．(3)は同期をとるタイミングに合わせてビデオ素材をあらかじめ分割する前処理（実際にビデオデータを分けるのではなく，構造の生成のみである）が必要になるが，それ以外は基本的な並列再生メカニズムで処理可能である．特に，絶対時間を意識せずに構造を記述できる点で，最も汎用性があり，利便性の高い方法と考えられる．

記述例の中にあるコンカチネーションフラグ(concatenation)は，論理構造上分割されているが，物理的には連続しているコンテントを示すため，オブジェクト記述の中に入れるフラグであり，先頭，中間，最終の区別を示す．以下に記述例を示す．

```
object {
  object-type   video,
  object-descriptor {
     object-identifier   0 0 0,
     user-visible-name
       "The first part of the cut",
     object "the_cut.cnt"
     concatenation   0   -- First
},
object {
  object-type   video,
  object-descriptor {
     object-identifier   0 0 1,
     user-visible-name
       "The second part of the cut",
     object "the_cut.cnt"
     concatenation   1   -- Middle
},
object {
  object-type   video,
  object-descriptor {
     object-identifier   0 0 2,
     user-visible-name
       "The last part of the cut",
     object "the_cut.cnt"
     concatenation   2   -- Last
}
```

4. 再生処理

本提案方式により記述されたビデオシーケンスを実際に再生するには，図5に示すようなフローにより再生処理を行う．初めに準備するものは，コンテントデータ，論理構造データ，再生属性データである．コン

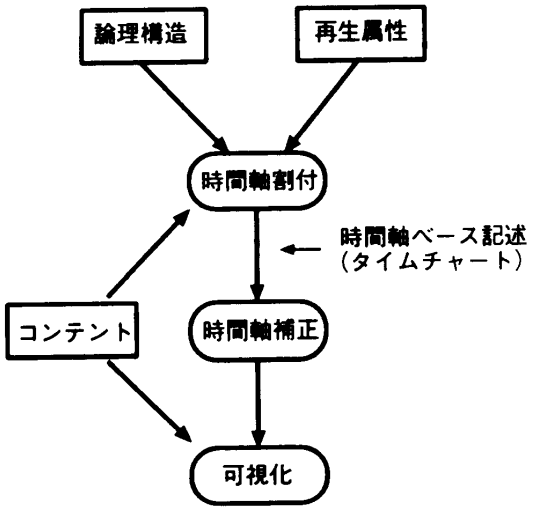

図5 再生処理フロー
Fig. 5 Presentation processing flow.

テントをカットチェンジでつなぐだけならば再生属性データは不要である．次に，これらのデータをもとに各コンテントの再生時間やオーバラップさせる時間などを判定して，各コンテントを再生/停止する時刻を記述した時間軸ベースの記述ファイル（タイムチャート）を生成する．さらに，実際のオーディオ・ビデオデータのディスクからの読み出し時間や復号処理の遅延など，実際の再生環境に依存するパラメータの補正を行った後，可視化を行う．

論理構造等からタイムチャートを作成する方法は各オブジェクトの再生処理をシミュレートすることによる．まず，構造記述ツリーの各オブジェクトにそれぞれ個別のプロセスを割り当て，マルチプロセスとして動作させる．このアプリケーションにおいて各プロセスは，論理構造が指定する時間軸上での並列関係，直列関係に従って，他のプロセスと同期をとりながら自律的に動作してコンテントの再生・停止等の制御動作をシミュレートする．ある時刻にこのアプリケーションを起動し，コンテントに対する制御動作を発生順にその時刻と内容を記録していくと，タイムチャートデータの要求する情報が得られる．

このようなモデルシステムを採用する理由は，動作の実現に必要な機能が単純であり，プロセス間の同期も親子関係にあるオブジェクトとのみとればよくローカルな状態だけに注目して処理を進めることができるためである．

アプリケーションが起動されると初めにPackageに対応するプロセスが起動され，各オブジェクトに割り当てられたプロセスは順にスタート待ち→処理中→終了の3状態を遷移する．「処理中」のプロセスは，実際には起動後すぐにその再生時間だけスリープさせ，親オブジェクトからの起動や，子オブジェクトの終了などのイベントが発生するとその時点でアクティブにして必要な処理を行い，次の処理までの待ち時間を指定して再びスリープ状態に戻る．

ここで，このシステムの動作は，時刻の記録のみが目的なので，実再生時間だけ待つ必要はない．すなわち，スリープ状態にあるオブジェクトの中から待ち時間が最も小さいものを選び，時刻を進めて次に実行すべきプロセス関数をただちに実行してもよい．この処理を繰り返し，スリープ状態のオブジェクトが何も残っていなければ処理を終了する．

なお，再生属性にフェードなどが設定されている場合には，その遷移時間を考慮してスリープ時間を調整し，オフセットが設定されている場合にはスリープ時間にその時間を加算する．

このように，論理構造からタイムチャートへの変換は容易かつ高速に行うことができる．

5. 評　　価

5.1 時間軸ベースの記述法との比較

シーケンサのように時間軸をベースにした構造記述（タイムチャート）は，自動プレゼンテーションシステムなどにおいて古くから用いられている．たとえば，ここで取りあげるMHEGに基づく時間軸ベース記述では，時間軸上にタイムストーンと呼ばれるマーカを置き，各々の時刻でコンテントの再生開始や停止の動作を行う．

MHEGでは「リンクオブジェクト」にアクションを起こすトリガ条件を記述しておき，条件が満たされるとリンクオブジェクトが参照している「アクションオブジェクト」が活性化される．アクションオブジェクトにはたとえばビデオの再生・停止などのアクションを記述しておく．同時に複数のアクションを起こすことが可能であるので，これにより複数のアクションの同期をとることができる．

時刻に同期してアクションを起こすには，所望の時刻に「タイムストーン」を設定しておく．時刻が設定されたタイムストーンをよぎるとイベントが発生し，あらかじめ規定されたアクションが起こる．**図6**にモデルを示す．

HyTime[11]においてもオブジェクト間の同期は時間軸上の位置を介して表現されるため，基本的にはMHEGと同様な記述手法となる．

上記から分かるように，時間軸ベースの記述は再生時の処理が容易であり，フェードやワイプなどのタイ

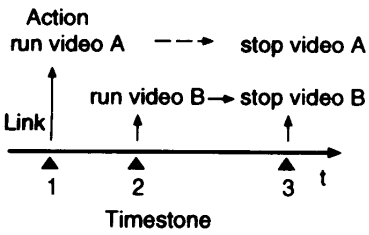

図6 MHEGによる同期記述モデル
Fig. 6 The synchronization description model based on MHEG.

表1 構造記述データ量と変換時間
Table 1 Structure description data amount and required conversion time.

シナリオ	LS	TC	CT
1	2.5 (10)	15.8 (5)	0.11
2	4.6 (20)	31.6 (9)	0.13
3	7.1 (30)	44.7 (11)	0.18
4	9.0 (40)	59.9 (16)	0.26
5	11.4 (50)	68.9 (16)	0.26
6	11.3 (50)	69.7 (19)	0.26
7	10.8 (50)	85.9 (29)	0.28
8	10.4 (50)	81.3 (33)	0.29

LS: 論理構造データ（KB）
（オブジェクト数）
TC: タイムチャート（MHEG）データ（KB）
（Timestone 数）
CT: 変換時間（sec.）

ミングも直接記述できるので簡便である利点がある．しかし，逆にコンテンツに依存する点，構造と修飾的情報が混在してしまう点，などで自動生成，再利用などが困難になる欠点がある．

本手法（Logical Structure: LS）およびMHEGに基づくタイムチャート記述（Time Chart: TC）の記述データ量の比較を行うと表1のようになった[12]．なお，各記述ファイルはASN.1値表記法に従ったクリアテキストで書かれている．データ量にはオーディオやビデオの実データは含まれていない．

シナリオ1から5までは論理構造のオブジェクト（ツリー構造のオブジェクト）を10ずつ追加していったもので，データ量もほぼオブジェクト数に比例して増大している．シナリオ5から8は，オブジェクト数を一定にして，並列再生が多く階層の深いものから直列再生のみの単純なものまで段階的にツリー構造を変化させて結果を比較したものである．これより，タイムチャートのデータ量は論理構造データのオブジェクト数よりもタイムストーン数に比例する傾向にあるといえる．論理構造データのサイズは1オブジェクトあたり約200 byte，タイムチャートデータのサイズは1タイムストーンあたり約3 kbyteであり，タイムチャートデータは，論理構造データと比較して6～8倍のデータ量となっている．

5.2 HyperODAとの比較

HyperODA[13]はODAをもとにして文書要素間のハイパリンクや同期関係を規定している．HyperODAでは本提案手法と同様に基本的には論理構造ベースの記述が行われる．相違点は，HyperODAでは論理構造の中でsynchronisation typeを指定することによって，ツリーの任意の分岐点で直列再生と，並列再生が選択できる点である．また，同じく論理構造の中に再生時間や再生開始時間などのコンテント依存データが入る点が異なる．

任意の構造が記述できる点は優れているが，反面タイムチャートへの変換処理が複雑になる欠点がある．またビデオの並列再生は実際上ハードウェアによる制限があり，多数並列にすることはできない．本提案でもツリー表現の中で構成要素の繰り返しを許すか，階層を拡張すれば十分に複雑な構造がとれるため，実際上は問題ないものと考えられる．

5.3 実時間再生

論理構造からタイムチャートデータへのデータ変換を，Sun Microsystems社製のSparc Station（SS10/51）を用いて行った．実験結果によると，論理構造データからタイムチャートデータへの変換に要する時間（Conversion Time: CT）はファイルの入出力処理時間も含めて約0.1～0.3秒であった．素片データ実体は論理構造データ，タイムチャートデータに含まれないため，素片データの長さは変換時間には影響しない．今回の実験で，オブジェクト数50，階層の深さ8のかなり複雑なツリー構造のシーケンスでも，瞬時にタイムチャートへの変換が行えることを確認した．したがって，蓄積にはコンパクトな論理構造データを用い，再生時にリアルタイムでタイムチャートへ変換を行って連続したビデオストリームを再生することも可能である．

6. むすび

ビデオシーケンスの論理構造に基づく記述方法を提案した．実験により論理構造記述からタイムチャートを生成し，それに基づいて実時間でビデオシーケンスの合成ができることも確認されている．本方法に対して並列に表示するオブジェクトを別々の位置に配置する場合の空間的割付に関するルールや属性を追加することにより，ビデオシーケンスばかりでなくマルチメディア一般，さらには三次元オブジェクトを含むシナリオの記述にも拡張可能と考えられる．

本論文では確定されたシナリオに基づくシーケンス

再生を扱った．今後可変速再生やストーリーの分岐をともなう対話型のシナリオの記述についても検討が必要になると考えられる．

謝辞 本研究は株式会社グラフィックス・コミュニケーション・ラボラトリーズにおいて行われたものである．本研究の機会を与えていただいた同社藤原 洋・研究開発本部長，ならびにご支援いただいた諸兄に感謝いたします．

参考文献

1) 浅見直樹，鈴木信夫，小林 修：マルチメディアOS生まれる，日経エレクトロニクス，No.534, pp.113–134 (1991).
2) 花村 剛，亀山 渉，富永英義：マルチメディア標準化のためのビデオ・ドキュメント・アーキテクチャの構想，信学技報，No.IE90-42, pp.21–28 (1990).
3) Weiss, R., Duda, A. and Gifford, D.K.: Composition and Search with a Video Algebra, *IEEE Multimedia*, Vol. Spring 1995, pp.12–25 (1995).
4) Hardman, L., van Rossum, G. and Bulterman, D.C.A.: Structured Multimedia Authoring, *ACM Multimedia 93*, ACM, pp.283–289 (1993).
5) ISO/IEC: ISO/IEC 8613-1 Information Technology – Open Document Architecture (ODA) and Interchange Format: Introduction and General Principles (2nd edition) (1994).
6) ISO/IEC: ISO/IEC 8879 Information Processing – Text and Office Systems – Standard Generalized Markup Language (SGML) (1986).
7) 滝川 啓：ビデオデータの構造化に関する一提案，第1回画像電子学会メディア統合技術研究会予稿，pp.67–74 (1993).
8) ISO/IEC: ISO/IEC CD 13522-1, Information Technology – Coded Representaion of Multimedia and Hypermedia Information objects (MHEG) – Part I: Base Notation (ASN.1) (1993).
9) ISO/IEC: ISO/IEC 8824 Information processing – Open Systems Interconnection – Specification of Abstract Syntax Notation One (ASN.1) (1990).
10) ISO/IEC: ISO/IEC 8825 Information Processing – Open Systems Interconnection – Specification of Basic Encoding Rules for Abstract Syntax Notation One (ASN.1) (1990).
11) ISO/IEC: ISO/IEC 10744, Information Technology – Hypermedia/Time-based Structuring Language (HyTime) (1992).
12) 権田亜紀子，滝川 啓，福留治隆：ビデオシーケンスの構造化とタイムチャートの生成，信学技報，No.IE95-23, pp.59–65 (1995).
13) ISO/IEC: ISO/IEC DIS 8613-14.2 Information Technology – Open Document Archtecture (ODA) and Interchange Format: Temporal Relationships and Non-linear Structures (1994).

(平成7年12月6日受付)
(平成8年5月10日採録)

滝川 啓（正会員）

昭和47年3月東工大工学部電気工学科卒業．同年4月電電公社（現NTT）武蔵野電気通信研究所入所．以来，横須賀研究所，HI研究所，画像通信事業本部等において画像通信システムの研究開発に従事．昭和56～57年，カナダ通信省通信研究所交換研究員．平成5～7年グラフィックス・コミュニケーション・ラボラトリーズへ出向し，マルチメディアシステムの研究に従事．平成7年4月よりNTTソフトウェアへ出向，マルチメディアシステムのSI等に従事．マルチメディア情報の構造化に興味を持つ．情報処理学会，テレビジョン学会，電子情報通信学会会員．

権田亜紀子

平成2年3月東京女子大学文理学部数理学科卒業．同年4月（株）日立製作所中央研究所入所．平成5年6月～平成7年5月（株）グラフィックス・コミュニケーション・ラボラトリーズ出向．ディジタルVTR用画像符号化，マルチメディアコンピューティングの研究に従事．平成7年9月より（株）日立製作所知的所有権本部に転属し，情報通信部門の知的所有権業務に従事．テレビジョン学会，電子情報通信学会会員．

福留 治隆

昭和26年生．昭和49年大阪大学基礎工学部生物工学科卒業．昭和52年同大学院修士課程修了．日本情報サービス（株）（現日本総合研究所（株））を経て，昭和59年（株）アスキーに入社．以来，プログラマとして，各種システム機器開発に従事．平成5年（株）グラフィックス・コミュニケーション・ラボラトリーズに出向．現在，主任研究員．