

構造化に基づくビデオシーケンスの合成

滝川 啓、佐藤 正人、植田 信吾、権田 亜紀子

グラフィックス・コミュニケーション・ラボラトリーズ

監視画像など特殊なものを除くと、ビデオシーケンスも文書と同様に論理構造を持っている。したがって、シーケンスの論理構造を記述することにより、各構成要素を成すビデオ素片を組み合わせて連続したシーケンスに合成することができる。

また、フェードやワイプなど再生効果を目的とした操作については、論理構造とは独立と考えられるので、別に記述することにより、編集の効率化や構造情報の再利用などが可能になる。

本稿では論理構造から再生タイミングを生成し、それを用いてリアルタイムに効果を含む連続したビデオシーケンスを合成するシステムについて、処理方式ならびにハードウェア構成を述べる。

Assembling Video Sequences with a Logical Structure

Kei Takikawa, Masato Sato, Shingo Ueda, Akiko Gonda

Graphics Communication Laboratories

Besides special cases such as surveillance, video sequences, as well as documents, have logical structures. If we can describe a logical structure of a video sequence, we can assemble a whole sequence with video data fragments.

Visual effects, such as "fade" or "wipe" are considered as independent attributes from logical structures. Then, describing visual effect attribute separately from a logical structure, editing efficiency and reusability of video data fragments are improved.

We have realized a realtime video sequence assembling technique, which includes a playback timing generation technique and a specially designed hardware.

まえがき

近年急速に普及しつつあるデジタルビデオ技術を用いることにより、従来のVTRやビデオディスクを前提としたビデオ応用システムとは根本的に異なるシステムが実現し得る。筆者らは昨年ビデオシーケンスの構造化を提案し(1)、これに基づいたシステムの実現を検討してきた。

提案する方式は、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ものである。このようにすることによって、編集時間、データ蓄積量の減少、あるいは構造のテンプレートをを用いたシーケンスの半自動生成の可能性、など種々の利点が生じる。

今般、構造化データに基づいて複数の素片データから連続したオーディオ・ビデオシーケンスをリアルタイムで合成することができたので報告する。

1. 論理構造に基づく構造化

1.1 アーキテクチャ

本研究では、オーディオ・ビデオシーケンスの入力・編集・再生に関して、文書処理のモデル(2)などに習って図1に示すようなフェーズ分けをして考える。入力フェーズは各カット(ここでは単位となる素片データ全般を指す)の録画や時間的・空間的トリミングのフェーズ、編集フェーズは各カット間の順序付けや組み合わせ、フェード・ワイプなど再生時の効果を決めるフェーズ、割付フェーズは各カットの長さや効果の持続時間などをもとに各カットの再生タイミングを決めるフェーズ、可視化フェーズは実際のシーケンスを生成するフェーズである。

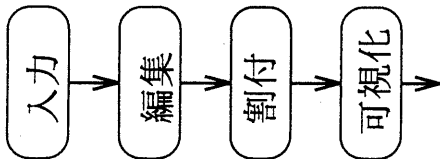


図1 文書処理のモデル

従来のビデオ編集等ではこれらが渾然一体となっており、編集作業に手間がかかる一因となっていた。本研究では、上記のようなモデルに基づいてシステムアーキテクチャを組み立てることにより、編集作業の短縮、さらには構造のテンプレートをを用いた自動編集(シーケンスの半自動生成)、などをねらいとしている。

図2にシステムアーキテクチャを示す。ここで、論理構造はカットの順序付け、組み合わせを記述したデータ、再生属性はフェード・ワイプなど再生時の効果を決めるデータ、コンテンツは各カットのデータ(素片データ)である。これらの関係を図3に示す。

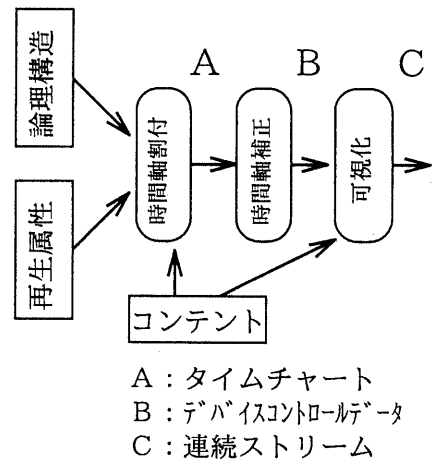


図2 システムアーキテクチャ

論理構造は「プロセッサブルフォーム」であり編集等に適しているが、具体的な時刻関係は分からないのでそのまま再生に用いることはできない。これに対して時刻関係を明示したデータをタイムチャートと呼ぶことにする。タイムチャートは「ファイナルフォーム」であり、編集等には向かない。

ユーザは第一に論理構造ならびに再生属性を決める。また、必要な素片データを用意する。次にこれらを時間軸割付処理に加え、各素片データの長さやフェード・ワイプ等の遷移時間を考慮して、時間軸上の開始・終了時間を決める。このデータをタイムチャートと呼ぶ。最後に、ディスクのシークタイムやオーディオ・ビデオデコーダのセットアップタイム(主としてバッファメモリへの初期データ

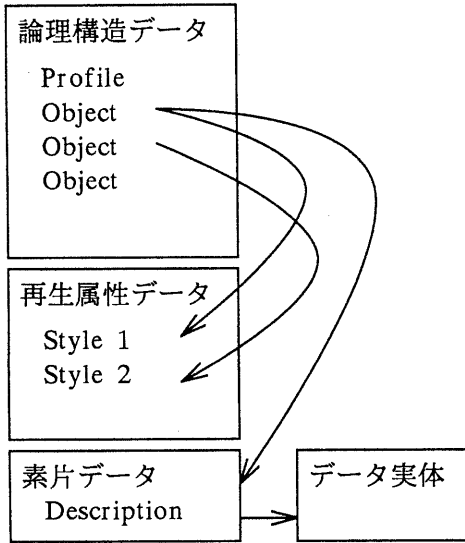


図3 各データの関係

書込時間)等を考慮して開始・終了時間を補正し、素片データを複数あるデコーダに割り付けて、最終的にデバイスを制御するデバイスコントロールデータを生成する。デバイスコントロールデータは個々のシステム特性に依存するので汎用性がない。データの蓄積や交換には、用途に応じて論理構造またはタイムチャートを用いる。

本方式の特徴は「編集」や「シーケンス合成」の際に、素片データ自体には全く変更を加えない点にある。したがって、素片データの利用効率が高く、処理時間も短い。

1. 2 論理構造

論理構造はオーディオ・ビデオシーケンスの意味的な構造、例えばイントロダクション、ボディ、エンディングなど、を表す。また、直列的なつながりだけではなく、オーバーラップ(ミックス)やはめ込みなど複数の素片データが並列に再生される構造も記述する。

構造の一般形を図4に示す。ここで、四角形は同一階層で並列に複数存在しうるオブジェクトを示し、角丸四角形はそれが直列に複数繰り返し得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持つオブジェクトであり、一重線のものは仮想的なオブジェクトである。

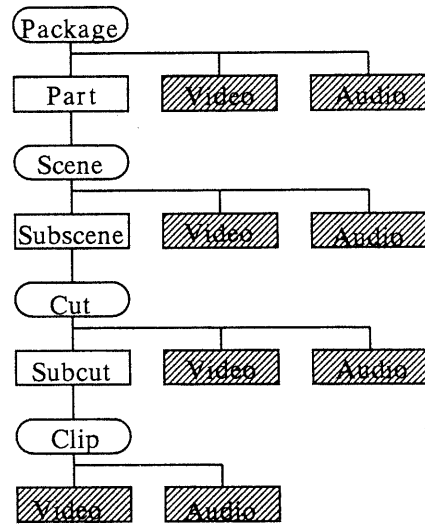


図4 論理構造の一般形

この一般形によれば並列に再生されるもの同士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ある。図5に具体的な論理構造のサンプルを示す。

また、データ構造はODA(2)等に習い、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構造としている。オブジェクトのASN.1記述例を以下に示す。

```
object {
  object-type scene,
  object-descriptor {
    object-identifier "0 0 0",
    user-visible-name "Introduction",
    subordinate {0, 1}
  },
}
object {
  object-type subscene,
  object-descriptor {
    object-identifier "0 0 0 0",
    user-visible-name "Dummy",
    subordinate {0, 1, 2},
    presentation-style-id {0}
  }
}
```

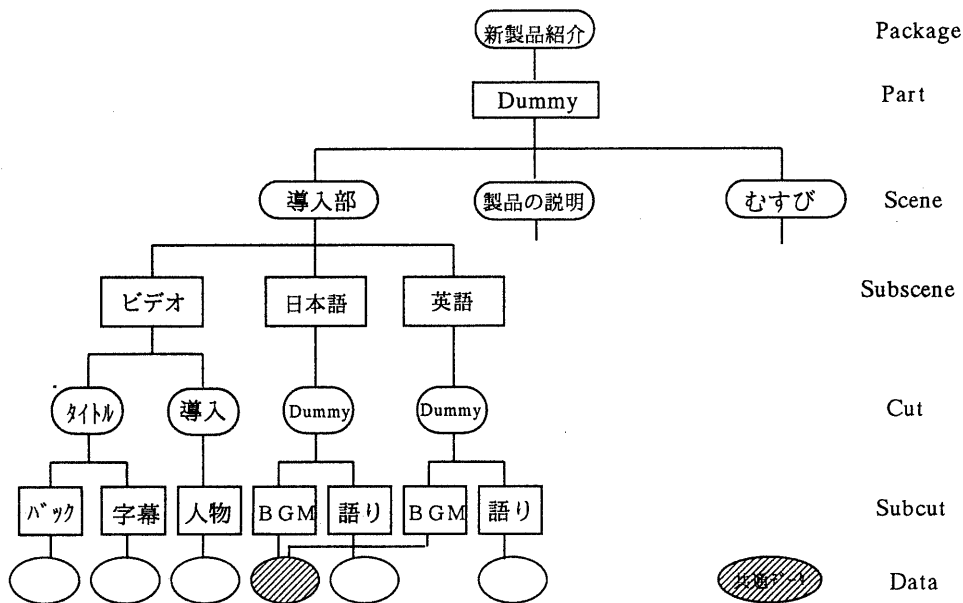


図5 構造の具体例

1.3 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再生時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。

- 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、オーバーラップ(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

再生属性データは論理構造データと独立しており、インデックス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。

データ記述の例を以下に示す。

```
linkage-style {
  style-id "0",
  linkage-directives {
    linkage {
      {object subordinate 0},
      {object subordinate 1},
      {object subordinate 2}
    },
    linkage-type cross-fade 2000
  }
}
```

1.4 コンテンツ

コンテンツはオーディオ・ビデオの実データの記述であり、構造はMPEG⁽³⁾のcontent-classをベースとしている。コンテンツは大別するとプロファイル部と実データ部に分かれ、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。また、

2. 3 アクション

=xx

2. 3. 1 基本形

タイムチャート上での各素片データの再生指示アクションは図7(1)のように表示の開始点でrunを、終了点でstopを実行する。

タイムストーン1 :

```
target A
run
```

タイムストーン2 :

```
target A
stop
```

2. 3. 2 フェードイン/フェードアウト

上記に再生属性としてフェードイン/フェードアウトを加える場合には、開始点のタイムストーンにおいて、runとフェードインの指定(遷移時間を含む)を行う。また、フェードアウトの開始点においてフェードアウトの開始と遅延時間を指定して表示の終了を実行する。図7(2)にタイミングを示す。

タイムストーン1 :

```
target A
run
set mixing on
set mixing level
level=100
transition-duration
```

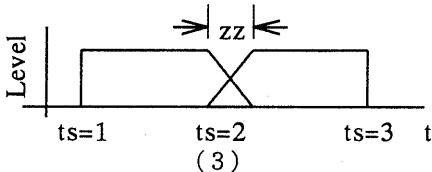
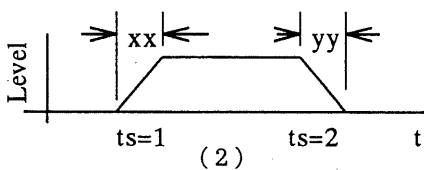
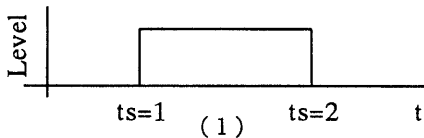


図7 アクションのタイミング

タイムストーン2 :

```
target A
delay yy; stop
set mixing on
set mixing level
level=0
transition-duration
=yy
```

2. 3. 3 クロスフェード

2つの素片をクロスフェードにより直列に結合する場合は、2番目のタイムストーンをクロスフェード開始点に置き、後の素片の開始、フェードの開始(遷移時間を含む)、遅延を指定した前のシーンの終了を指定する。タイミング関係は図7(3)の通りである。

タイムストーン1 :

```
target A
run
```

タイムストーン2 :

```
target A
delay zz; stop
set mixing on
set mixing level
level=0
transition-duration
=zz
```

target B

```
run
```

```
set mixing level
```

```
level=100
```

```
transition-duration
```

```
=zz
```

タイムストーン3 :

```
target B
```

```
stop
```

3. シーケンスの生成

3. 1 ハードウェア

本方式によりリアルタイムでシーケンスの合成を行うためには特別にデザインされたハー

ドウェアが必要である。実験システムのハードウェア構成を図8に示す。システムはオーディオ・ビデオのデータを蓄積するハードディスク、複数のデコーダとエフェクタからなり、それらをワークステーションからコントロールできるようにした。ビデオ符号化はMotion JPEGを適用している。また、エフェクタは市販のデジタルエフェクタを改造して用いた。2台のビデオデコーダは同期運転されている。

これらをタイムチャートに従ってコントロールすることにより、種々の再生パターンを作り出すことができる。

3.2 デバイスドライブデータ

本システムではリアルタイムにシーケンスの合成を行うので、各素片データの再生が同期しており、間隙なしに行われる必要がある。オブジェクト間の正確な同期を取るためには各種の処理遅延時間を考慮する必要がある。また、各素片データを複数あるデコーダのどれに割り付けるかを決定する必要がある。デバイスドライブデータは、図9に示すようにハードディスクドライブのシークタイム等の各種遅延要素を考慮すると共に、最小の処理コストで各デバイスをコントロールできるようにしたデータである。したがって、データ値はシステムを構成する機種に依存する。

デバイスドライブデータは、以下に示すよ

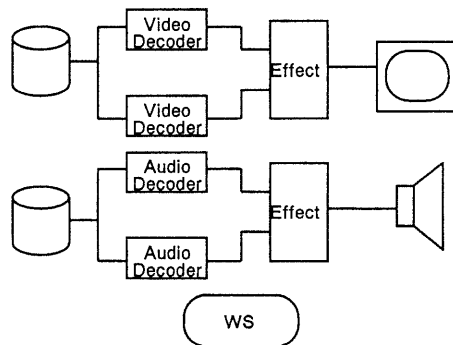


図8 ハードウェア構成

うに時刻毎のアクション内容を逐一記述したものである。最小時間分解能は約10msである。また、前方・後方にスキップサーチできるように、スキップ対象の時刻にマーカをつけている。

フォーマット

種別+時刻、コマンド種別、チャンネル番号、arg1、arg2 ---- ;

サンプル

T0, V, 0, 10, intro;	ビデオ再生
T100, VE, 10, 100, ---;	ビデオ効果
S300;	スキップ
	マーカー
T400, A, 3, 00	オーディオ再生

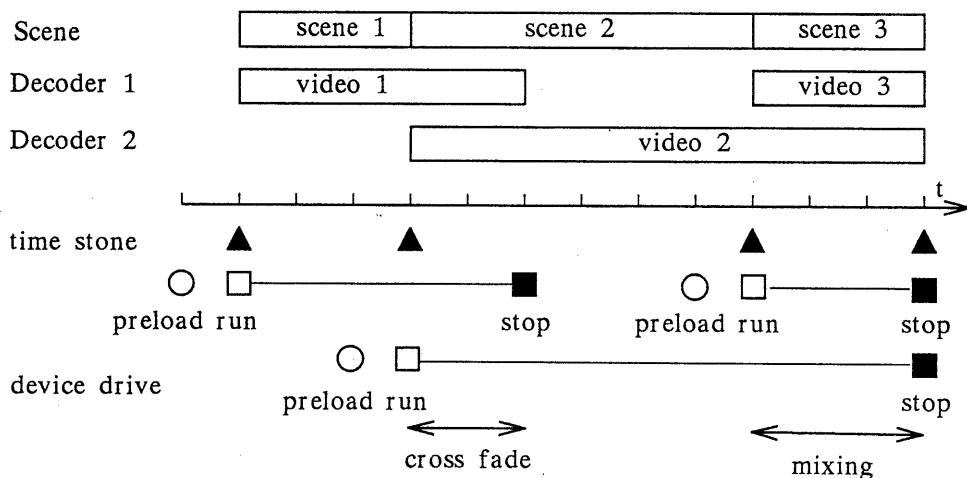


図9 デバイスドライブデータ

シーケンスの合成時にはタイムチャートからデバイスドライブデータを生成し、これにより各デバイスを制御する。

4. 実験結果

4.1 記述ファイルのデータ量

各記述ファイルはASN.1バリュー表記法によりクリアテキストで書かれている。サンプルのデータ量を表1に示す。オーディオやビデオの実データは含まれていない。

表1 記述ファイルのデータ量

シナリオ	データ量 (ライン)	
	構造化データ (オブジェクト数)	タイムチャートデータ (タイムストーン数)
1	431(15)	829(4)
2	1001(30)	1946(8)
3	1643(49)	3029(13)

論理構造ファイルのサイズは概ねオブジェクトの数に比例しており、1オブジェクト当たり約30ラインである。一方、タイムチャートファイルのサイズは概ねタイムストーンの数に比例しており、1タイムストーン当たり約200ラインである。また、オブジェクト数とタイムストーン数の関係はおおよそ4:1である。このように、記述ファイルとしては論理構造記述の方がコンパクトである。また、いずれの場合もオーディオ・ビデオの実データに比べれば無視できる量である。

4.2 処理時間

デバイスドライブデータの解釈と各機器の制御はソフトウェアで行っているが、実用上十分な速度が得られている。また、タイムチャートを与えてからビデオが再生開始されるまでの時間はおよそ0.5秒である。遅れの要因はハードディスクのシーク時間、デコーダのバッファへの初期書込時間、GUI処理時間等である。

オブジェクト単位のスキップ再生の場合も同じレスポンスであり、ビデオディスクに比

べるとやや遅いが、VTRに比べれば十分早い。また、素片の接続時にはデバイスドライブデータにより、次の素片データを予め読み出してデコーダにプリロードするので、素片間でデータが途切れることはない。

論理構造からタイムチャート、タイムチャートからデバイスドライブデータへの変換に要する時間は上記の遅延時間に比べれば、無視できる値である。

むすび

構造化されたオーディオ・ビデオシーケンスのリアルタイム再生を実現した。本方式を適用することによりビデオ素片のデータベース化とシーケンスの半自動合成が可能になるほか、シーケンス編集の省力化が期待できる。また、シーケンスの再編集が必要な場合に、論理構造+再生属性+コンテンツの形態でのデータ交換や通信も可能になる。

今後、これらの応用システムの検討を進めて行きたい。

参考文献

- (1) 滝川, "ビデオデータの構造化に関する一提案", 画像電子学会メディア統合技術研究委員会第1回研究発表会予稿, pp.67, 1993.9
- (2) ISO, "Information processing - Text and office systems - Office Document Architecture(ODA) and interchange format - part 1: Introduction and general principles", ISO 8613-1:1989(E), pp.20, 1989
- (3) ISO, "Information Technology - Coded Representation of Multimedia and Hypermedia Information objects (MHEG) - Part I: Base Notation (ASN.1)", Committee Draft, ISO/IEC CD 13522-1, 1993