동적 비디오 객체 데이타 모델(DVID)

(Dynamic Video Object Data Model(DVID))

송 용 준 † 김 형 주 ††
(Yong-Jun Song) (Hyoung-Joo Kim)

요 약 이제까지 비디오 데이타베이스를 모델링하기 위한 많은 연구들이 수행되었지만 그 모든 모델들에서 다루는 비디오 데이타는 사용자의 개입이 없을 때 항상 미리 정의된 순서로 보여진다는 점에서 정적 데이타 모델로 간주될 수 있다. 주문형 뉴스 서비스, 주문형 비디오 서비스, 디지털 도서관, 인터넷 쇼핑 등과 같이 최신 비디오 정보 서비스를 제공하는 비디오 데이타베이스 응용들에서는 빈번한 비디오 편집이 요구되는데 실시간 처리가 바람직하다. 이를 위해서 기존의 비디오 데이타 내용이 변경되거나 새로운 비디오 데이타가 생성되어야 하지만 이제까지의 비디오 데이타 모델에서는 이러한 비디오 편집 작업이 일일이 수작업으로 수행되어야만 했다. 본 논문에서는 비디오 편집에 드는 노력을 줄이기 위해서 객체지향 데이타 모델에 기반하여 DVID(Dynamic Video Object Data Model)라는 동적 비디오 객체 데이타 모델을 제안한다. DVID는 기존의 정적 비디오 객체뿐만 아니라 사용자의 개입없이도 비디오의 내용을 비디오 데이타베이스로부터 동적으로 결정하여 보여주는 동적 비디오 객체도 함께 제공한다.

Abstract A lot of research has been done on modeling video databases, but all of them can be considered as the static video data model from the viewpoint that all video data on those models are always presented according to the predefined sequences if there is no user interaction. For some video database applications which provides with up-to-date video information services such as news-on-demand, video-on-demand, digital library, internet shopping, etc., video editing is requested frequently, preferably in real time. To do this, the contents of the existing video data should be changed or new video data should be created, but on the traditional video data models such video editing works should be done manually. In order to save trouble in video editing work, this paper proposes the dynamic video object data model named DVID based on object oriented data model. DVID allows not only the static video object but also the dynamic video object whose contents are dynamically determined from video databases in real time even without user interaction.

1. 서 론

오늘날 멀티미디어 데이타는 주문형 뉴스(NOD) 서비스, 주문형 비디오(VOD) 서비스, 인터넷 쇼핑, 디지털 도서관 등 여러 응용 분야에서 중요하게 사용되고 있지만 관련 기술들의 수준에는 아직도 발전의 여지가 많다. 그러한 분야들 중의 하나가 데이타의 합성, 표현, 저장 및 관리 등에 관한 기반을 제공하는 멀티미디어 데이타 모델로서 이제까지 많은 연구들이 수행되어 왔지만 그 각각이 장단점을 가진 채 아직까지는 표준적인 모델이 없는 실정이다.

특히 가장 복잡하고 비정형적인 멀티미디어 데이타 타입인 비디오에 대해서도 많은 연구들이 수행되어 왔다. AI 분야에서 Allen이 제안한 간격기반 시간 논리(interval-based temporal logic)[1]를 기반으로 Little 등이 제안한 간격기반 시간 합성 모델[2]과 같은 시간 기반 비디오 모델을 비롯하여, 각 비디오를 비디오 세그먼트들의 대수적(algebraic) 결합으로 표현한 Weiss 등의 대수적 비디오 모델[3], 비디오 객체들을 이용하여 비디오를 표현하는 Oomoto 등의 비디오 객체 데이타 모델[4]과 멀티미디어 객체들을 위한 클래스 계층과 그

† 비 회 원 : 서울대학교 컴퓨터공학과
yjsong@oopsla.snu.ac.kr
†† 비 회 원 : 서울대학교 컴퓨터공학과 교수
hjk@oopsla.snu.ac.kr
논문접수 1998년 5월 26일
심사완료 1999년 7월 19일

것에 기반하여 멀티미디어 객체를 합성하기 위해 Gibbs 등이 제안한 복합 멀티미디어 객체 모델[5], 그리고 비디오 정보의 공유에 중점을 두어 비디오 정보의 일반적인 특성들을 각각 독립적으로 표현하는 Hjelsvold 등에 의한 VideoSTAR 데이타 모델[6] 등 여러 연구들이 수행되어 왔다. 이와 같은 기존의 비디오 데이타 모델 연구들은 모두 비디오 데이타의 구성 요소들을 정의하고 그것들을 시간적 그리고/또는 공간적으로 합성하여 새로운 비디오 데이타로 생성하는 방법[7]과 비디오 인덱싱 및 검색 기법에 주된 관심을 두고 있다[6,8,9]. 여기서 이 모델들에 의해서 생성된 모든 비디오 데이타는 미리 정해진 고정된 재생 순서를 가지고 있어 사용자 개입이 없다면 항상 같은 내용을 보여주므로 본 논문에서는 정적 비디오 데이타라 하며 기존의 모델들을 정적 비디오 데이타 모델이라 한다.

일반적으로 사용자들에게 최신의 비디오 정보 서비스를 제공하는 비디오 데이타베이스 응용들에서는 비디오 편집 작업이 빈번하게 발생한다. 예를 들면, NOD에서 사용자가 원하는 특정 인물에 관련된 뉴스 비디오를 최근 것부터 일정 시간 분량으로 편집하여 보여주거나[8], VOD에서 가장 인기있는 비디오들, 최신 출시 비디오들 등 여러 가지 최신 카탈로그 서비스를 제공하는 것이 필요하며, 마찬가지로 인터넷 쇼핑 서비스에서도 가장 인기있는 상품들, 최신 추천 상품들 등과 같은 상품 정보 카탈로그들을 최신 비디오 정보로 자주 갱신하는 것이 필요하다. 디지털 도서관에서는 비디오를 포함한 여러 타입의 새로운 정보를 지속적으로 추가하는 것이 필요하다. 하지만 기존의 정적 비디오 데이타 모델에서는 이러한 비디오 편집 작업들을 일일이 수작업으로 수행해야하기 때문에 많은 시간과 인적 노력이 요구되는데, 이를 줄이기 위해서 본 논문에서는 객체지향(OO) 데이타 모델에 기반한 동적 비디오 객체 데이타 모델 DVID(Dynamic Video Object Data Model)를 제안한다. OO 데이타 모델은 추상 데이타 타입, 복합 객체, 객체 식별자 등 멀티미디어 데이타를 모델링하는 데 유용한 특성들을 제공하기 때문에, 객체지향 데이타베이스 관리 시스템(OODBMS)은 가장 유망한 멀티미디어 DBMS의 하나로 손꼽히고 있으며[10], 본 논문에서 제안하는 DVID 기반의 비디오 데이타베이스 시스템을 OODBMS 상에 용이하게 개발할 수 있다. 비디오 데이타 모델 분야에서의 본 논문의 기여는 다음과 같다.

1) OO 데이타 모델을 기반으로 간결한 비디오 데이타 모델을 제안하여 비디오 편집 작업을 보다 용이하게 지원한다. 제안된 모델에서 기존의 객체들을 이용하여 복합 객체 형태의 새로운 비디오 객체를 쉽게 생성할 수 있으며, 객체 식별자를 이용하여 데이타 중복없이 각 객체가 공유될 수 있어 디스크 공간을 절약할 수 있을 뿐만 아니라 데이타 일관성도 제공한다.
2) 기존의 정적 비디오 객체뿐만 아니라 동적 비디오 객체를 제안하여 수작업에 의한 비디오 편집 작업을 부분적으로 제거할 수 있다. 동적 비디오 객체는 실제로 보여질 때 지정된 조건의 처리 결과에 따라서 그 내용이 동적으로 결정된다. 이와 같은 동적 비디오 객체를 이용하면 비디오 데이타베이스에 대한 특정 종류의 변경 사항들을 관리자의 작업없이도 자동으로 반영할 수 있다.

본 논문의 구성은 다음과 같다. 먼저 2장에서는 DVID의 기본 개념들을 소개하고, 3장에서는 DVID를 보다 정형적으로 정의한다. 4장에서는 DVID의 실제 구현을 위한 각 비디오 객체 타입들의 논리적 설계 및 처리에 대하여 설명하며, 5장에서 DVID의 구현을 위해 성능 문제를 고려한다. 6장에서 동적 비디오 객체 방식과 기존의 질의 저장 방식의 차이를 설명하고, 마지막으로 7장에서 본 논문의 요약과 향후 연구 과제를 언급하며 결론을 내린다.

2. 동적 비디오 객체 데이타 모델의 기본 개념

본 연구에서는 기존의 비디오 정보를 공유하며, 그로부터 새로운 비디오 정보를 쉽게 생성하고 비디오 편집을 보다 쉽게 지원하도록 비디오 데이타베이스를 모델링하는 것을 목적으로 한다. 일반적으로 비디오 데이타베이스 시스템은 사용자가 원하는 비디오를 검색하여 선택된 비디오에 대하여 다양한 조작을 할 수 있도록 하며, 특정 사용자에게는 비디오 검색에 사용될 비디오 특징 추출이나 주석 작성 등의 비디오 인덱싱을 수행할 수 있도록 한다. 각 비디오는 샷(shot), 시퀀스(sequence), 장면(scene)들로 구성된 마스터 DB를 가지고 있어서 사용자는 질의를 통해 DB 상의 관련된 장면들을 검색할 있고, 새로운 비디오 DB를 생성하여 기존의 비디오 DB와 상호 연결할 수도 있다[11]. 이와 같은 비디오 데이타베이스 시스템을 개발하기 위해서는 비디오를 자연스럽게 표현하는 동시에 비디오 편집, 인덱싱 및 검색이 효율적으로 수행할 수 있도록 비디오 데이타를 유연하게 모델링하는 것이 기본적으로 필요하다. 본 연구에서는 OO 데이타 모델에 기반하여 다음과 같은 기본 개념들을 갖는 비디오 데이타 모델을 제안한다.

먼저 비디오 인덱싱에 있어서 데이타의 시간성을 고려하는 것이 중요하다. 이를 위해서 비디오 데이타를 분할하는 세그먼트화 기법(segmentation)과 임의의 구간들을 선택하는 계층화 기법(stratification)이 제안되었는데, 후자는 일반화된 계층화 기법(generalized stratification)으로 개선되었다[9]. 일반화된 계층화 기법은 여러 시간 간격들을 하나의 계층으로 묶어 인덱싱을 하는 방식으로 계층화 기법에 비해 유연하지만 각 계층이 일일이 수작업으로 지정되어야 하고 일반화된 계층들 사이의 관계를 전혀 고려하지 않는다는 단점이 있다. 본 연구에서는 비디오 분할 기법을 이용하여 자동으로 검출한 세그먼트 각각을 계층의 단위로 사용하고, 각 계층이 세그먼트뿐만 아니라 다른 계층도 포함할 수 있도록 일반화된 계층을 확장함으로써 계층들 사이의 시간 관계를 보다 유연하게 표현할 수 있다. 이러한 인덱싱 방법은 비디오 데이타를 자동으로 분할하며, 분할에 사용된 비디오 특성을 이용한 내용 기반 비디오 검색과 각 계층마다 해당 비디오 정보를 표현하는 주석을 이용한 주석 기반 비디오 검색을 함께 사용하는 비디오 인덱싱 및 검색이 가능하다는 장점이 있다.

DVID에는 기본적으로 장면(scene) 비디오 객체와 파생(derived) 비디오 객체라는 두가지 정적 비디오 객체가 있다. 비디오에서의 장면이란 시공간적으로 연속된 동작을 보여주는 연속된 비디오 프레임들을 의미하는 것으로서[6] 원시 비디오에 대한 비디오 분할의 결과로 얻어진 각 세그먼트마다 하나의 장면 비디오 객체가 생성된다. 이 타입의 객체는 세그먼트 데이타, 분할에 사용된 특징 데이타 및 주석으로 구성되며, 사용자에게 주석 정보가 추가된 장면으로 보여지게 된다. 파생 비디오 객체는 장면 비디오 객체나 다른 파생 비디오 객체 등 기존의 비디오 객체들을 원소로 하는 비디오 객체 리스트와 해당 비디오 정보를 표현한 주석으로 구성되며, 사용자에게는 리스트 내의 원소들이 하나의 연속된 비디오로 보여지게 된다. 이 때, 원시 비디오로부터 생성된 모든 세그먼트들을 순서대로 리스트에 포함하는 파생 비디오 객체는 원래의 원시 비디오와 같은 내용으로 보여지는데, 그러한 객체를 특별히 기본(primary) 비디오 객체라 한다.

때때로 파생 비디오 객체에 대한 갱신이 필요할 수 있다. 기존의 비디오 모델에서는 이를 위한 비디오 편집 작업을 일일이 수작업으로 처리해야 하므로 만일 자주 발생하는 경우라면 아주 귀찮고 지루한 작업이 될 것이다. 하지만 모든 비디오 편집 작업을 자동으로 처리하기는 불가능할지라도 일정한 유형을 갖는 몇가지 변화는 자동으로 처리될 수 있다. 본 논문에서는 파생 비디오 객체에 대한 자동 처리 가능한 변화 유형으로 다음과 같은 세가지를 제안한다.

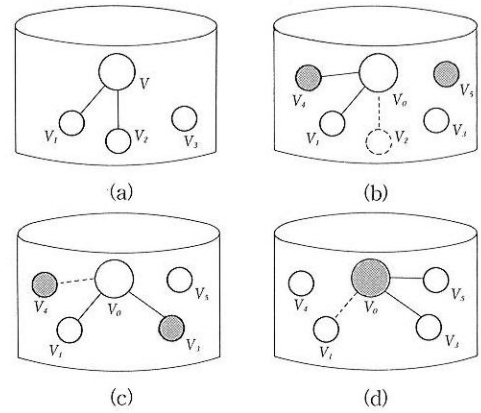

그림 1 비디오 객체의 변화 유형

1) 비디오 데이타베이스의 변화에 따른 파생 비디오 객체의 변화이다. 파생 비디오 객체의 생성 정의에 따라 포함되어야만 하는 비디오 객체가 새로 추가되거나 파생 비디오 객체 내에 포함되어 있던 요소 비디오 객체가 데이타베이스로부터 삭제되는 경우 그 파생 비디오 객체에서도 적절한 변화가 있어야만 한다. 예를 들어, 그림 1-(a)에서와 같이 비디오 객체 V_1, V_2를 차례로 포함하는 파생 비디오 객체 V_0을 가정하자. 새로운 비디오 객체 V_4, V_5가 추가되었을 때, 만일 V_4가 V_0의 생성 정의를 만족하는 특징을 갖는다면 V_4는 V_0에 포함되어야 하며, 기존의 V_2가 데이타베이스에서 삭제되었을 때에는 V_0의 구성 요소에서 삭제되어야만 한다(그림 1-(b) 참조).
2) 요소 비디오 객체의 상태 변화에 따른 파생 비디오 객체의 변화이다. 기존 요소 비디오 객체의 상태 변화로 인해서 더이상 파생 비디오 객체의 요소가 될 수 없거나 그와는 반대로 특정 비디오 객체의 상태가 변함에 따라 새로이 파생 비디오 객체의 요소로 포함되어야 하는 경우가 있다. 예를 들어 그림 1-(b)의 V_3과 V_4의 상태가 변경되어 각각 V_0의 생성 정의를 만족하고, 만족하지 않게 되었다면, 그 결과로 V_0은 그림 1-(c)에서와 같이 V_1, V_3을 포함한 객체로 변경되어야 한다.
3) 파생 비디오 객체 자체의 정의에 대한 변화이다. 때때로 관리자는 기존의 파생 비디오 객체가 다른

내용을 보여주도록 변경할 수 있는데, 이를 위해 원하는 내용의 비디오 객체들을 포함하도록 파생 비디오 객체의 생성 정의를 변경해야만 한다. 예를 들어, 관리자가 그림 1-(c)의 V_0이 V_5를 포함하도록 생성 정의를 변경했더니 기존 요소인 V_1은 그 정의에 부합하지 않아서 V_0으로부터 삭제되어야만 하는 경우이다(그림 1-(d) 참조).

이와 같은 변화 유형들을 자동으로 처리하기 위해서 본 논문에서는 내용에 사용될 가능성이 있는 후보 요소 비디오 객체들의 집합과 그로부터 실제 사용되는 요소 객체들을 결정하기 위한 조건으로 구성된 동적 비디오 객체를 제안한다. 정적 비디오 객체는 생성될 때 포함하는 요소를 결정하여 항상 고정된 내용을 보여주는 것에 비해 동적 비디오 객체는 사용자에게 보여질 때 후보 요소 객체들 중에서 지정된 조건을 만족하는 것들이 요소 객체가 되기 때문에 사용자에게 보여지는 내용은 비디오 데이타베이스의 상태에 따라 달라질 수 있다. 그림 2에서는 후보 요소 비디오 객체들의 집합 Ssv={V_1, V_2, V_3, V_4, V_5}와 요소 결정 조건 Cs로 구성된 동적 비디오 객체 D_v가 처리될 때의 상태에 따라 여러 가지 파생 비디오 객체들로 보여질 수 있다는 개념을 보여준다.

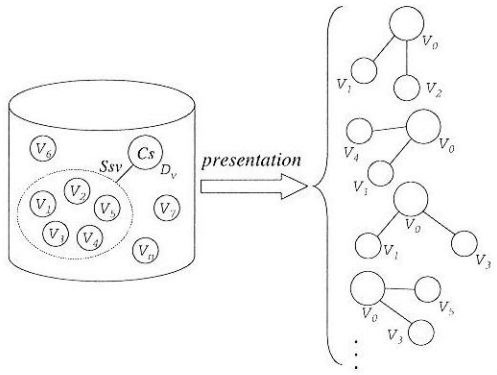

그림 2 비디오 객체의 동적 표현

3. DVID의 정형적 정의

DVID는 크게 나누어 장면 비디오 객체, 파생 비디오 객체, 그리고 동적 비디오 객체의 세가지 비디오 객체 타입을 정의한다. 각 타입은 비디오 내용뿐만 아니라 제목, 생성 시각, 접근 횟수 등과 같은 기본적인 비디오 정보를 위한 시스템 정의 속성들과 그 외의 임의의 정보를 저장하기 위한 사용자 정의 속성을 포함한다.

먼저 장면 비디오 객체(Ocv)는 본 연구에서의 비디오 처리 기본 단위로 사용되는데, 하나의 Ocv는 원시 비디오에 대한 비디오 분할의 결과로 얻어진 각 세그먼트의 비디오 프레임들 및 관련 정보들을 포함한다. 어떤 비디오 응용에서는 보다 세밀한 정보를 위해 프레임 단위의 객체 타입이 필요할 수도 있지만 그러기 위해서는 엄청난 수의 객체들이 추가로 필요하기 때문에 비효율적일 수 있다. 더구나 각 프레임은 장면 데이타 내에서의 위치, 프레임의 타입과 같은 논리적 정보만으로도 쉽게 접근하고 처리할 수 있기 때문에 본 연구에서는 물리적 프레임 객체 타입을 고려하지 않는다.

정의1. Ocv는 (Oid, Lfr, Vs, Vu)로 정의된다.

1) Oid는 데이타베이스 내에서 유일한 값을 갖는 객체 식별자이다.
2) Lfr은 한 장면을 구성하는 연속된 프레임들의 물리적 데이타이다.
3) Vs는 시스템 정의 속성으로서 속성 a_i와 그 값 v_i, $(1 \le i \le n)$로 이루어진 n-튜플 $<a_1 : v_1, a_2 : v_2, \ldots, a_n : v_n>$ 형식을 갖는다. 각 속성은 모든 객체에 필수적으로 포함되는 것으로서 문자열 타입의 비디오 제목, 정수 타입의 프레임 수, 문자열 집합 타입의 키워드, 날짜 타입의 생성 시각, 정수 타입의 접근 횟수 등이 정의된다.
4) Vu는 사용자 정의 속성으로서 Vs와 유사한 n-튜플 $<a_1 : t_1 : v_1, a_2 : t_2 : v_2, \ldots, a_n : t_n : v_n>$ 형식을 갖지만 사용자가 원하는 모든 정보를 저장할 수 있도록 각 속성마다 데이타 타입 t_i, $(1 \le i \le n)$가 추가된다. t_i로는 정수, 문자열 등의 기본 타입과 정의된 비디오 객체 타입들(또는 각각의 집합 타입들)이 사용될 수 있으며, 각 속성에 대한 값 v_i는 다음과 같이 정의된다.
   - 정수, 문자열로 표현되는 각 원소값(atomic value)은 값이다.
   - 각 비디오 객체의 Oid는 값이다.
   - 각 값 $v_1, \ldots, v_m$에 대하여 그 집합 {$v_1, \ldots, v_m$} 또한 값이다.

다음으로 파생 비디오 객체(Orv)는 데이타베이스에 저장된 비디오 객체들로 구성된 리스트를 포함하여 그 원소 순서대로 보여주는 것으로서 기존의 비디오 객체들을 보여주는 새로운 비디오 객체를 생성하는데 유용하다.

정의2. Orv는 (Oid, Lov, Vs, Vu)로 정의된다.

1) *Oid, Vs, Vu*는 *Ocv*에서와 동일하다.
2) *Lov*는 $[Ov_1, Ov_2, ..., Ov_n]$ 형식의 비디오 객체 리스트이다. Ov_i, $(1 \le i \le n)$로는 *Ocv*, *Orv* 또는 다음에 정의될 동적 비디오 객체가 사용된다.

앞에서 정의한 *Ocv*와 *Orv*는 각각 *Lfr*와 *Lov*의 원소 순서대로 항상 고정된 비디오 내용을 보여주는 정적 비디오 객체이다. 그와는 달리 동적 비디오 객체(*Odv*)는 그 내용이 실제로 보여질 때 동적으로 결정된다는 특징을 갖는다.

정의3. *Odv*는 (*Oid, Ssv, Vs, Vu, Cs*)로 정의된다.
1) *Oid, Vs, Vu*는 *Ocv*에서와 동일하다.
2) *Ssv*는 후보 요소 비디오 객체들의 집합으로서 $\{Osv_1, Osv_2, ..., Osv_n\}$ 형식을 갖는데, Osv_i, $(1 \le i \le n)$는 정적 비디오 객체로서 *Ocv* 또는 *Orv*가 될 수 있다. *Ssv*가 공집합인 경우에는 모든 정적 비디오 객체들이 후보 요소 비디오 객체로 처리된다.
3) *Cs*는 *Ssv*로부터 실제 사용될 요소 비디오 객체들을 순차적으로 선택하기 위한 요소 결정 조건으로서 다음과 같이 정의된다.
   - $x\ \theta\ c$로 표현되는 원소조건은 *Cs*이다. *x*는 속성, *c*는 상수값 또는 상수값을 생성하는 산술식, 그리고 θ는 $=, <, \le$(또는 그들의 부정인 $\neq, >, \ge$) 중의 하나이다.
   - $X \perp Y$는 *Cs*이다. *X, Y*는 *Cs*이며, $\perp$는 논리 연산자 $\wedge, \vee, \neg$ 중의 하나이다.
   - $X \Uparrow x$ 또는 $X \Downarrow x$는 *Cs*이다. *X*는 *Cs*, *x*는 속성, $\Uparrow$와 $\Downarrow$는 정렬 기준으로서 각각 검색 결과를 *x*에 대한 오름차순과 내림차순으로 정렬한다. 만일 정렬 기준이 생략된 경우에는 임의 정렬을 가정한다.

4. DVID에서의 비디오 객체 처리

4.1 비디오 객체 타입을 위한 클래스 설계

비디오 클래스(Video)는 그 내용의 고정성에 따라 크게 정적 비디오 클래스(StaticVideo)와 정적 비디오들의 집합(SetSVideo)을 포함하는 동적 비디오 클래스(DynamicVideo)로 나눌 수 있으며, 정적 비디오 클래스는 다시 장면 비디오 클래스(SceneVideo), 장면 비디오의 리스트(ListCVideo)를 포함하는 기본 비디오 클래스(PrimaryVideo), 그리고 임의 타입의 비디오들의 리스트(ListVideo)를 포함하는 파생 비디오 클래스(DerivedVideo)로 나눌 수 있다. 이와 같은 비디오 클래스들을 객체지향 기법에서의 IS-A 관계와 IS-PART-OF 관계에 따라 클래스 계층도로 표현하면 그림 3과 같다.

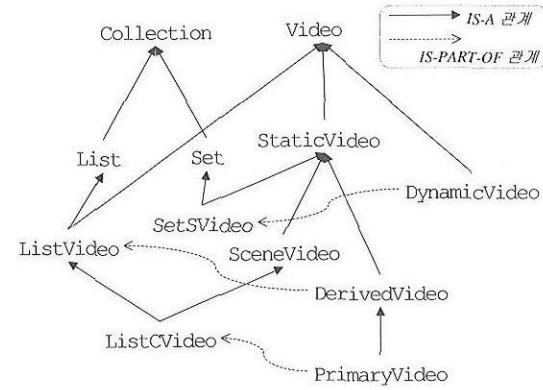

그림 3 비디오 객체들을 위한 클래스 계층도

4.2 원시 비디오 데이타 처리

원시 비디오 데이타의 처리 예로서 그림 4에서 세개의 장면 비디오 객체(*Ocv*)들을 생성하고, 그로부터 하나의 기본 비디오 객체(*Opv*)를 생성하는 과정을 보여준다.

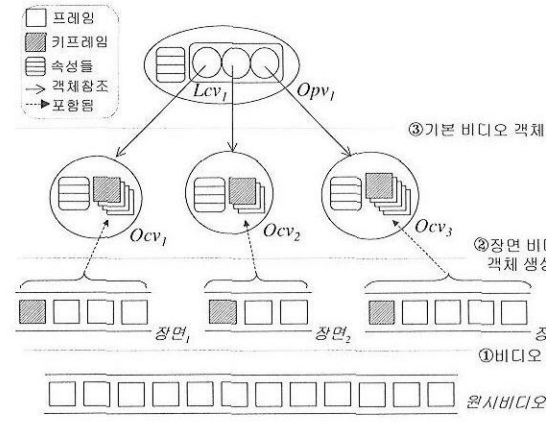

그림 4 원시 비디오 데이타 처리

먼저 원시 비디오 데이타를 분석하여 여러 장면들로 분할하는 과정이 필요하다. 이러한 비디오 분할 방법으

로는 크게 나누어 픽셀/블럭 비교법, 히스토그램 비교법, MPEG 비디오의 DCT 계수 이용법, 그리고 부분대역 특징 비교법 등이 있는데, 어떤 방법을 사용할 것인지는 구현 문제이므로 여기서는 더이상 언급하지 않기로 하고, 각 비디오 분할 방법에 대한 자세한 설명은 [12,13]을 참조하기 바란다.

다음으로 비디오 분할 과정에 의해 검출된 각 장면마다 연속된 해당 프레임들을 포함하는 장면 비디오 객체 (Ocv_i, i = 1, 2, 3)를 생성하고, 각 객체 내의 시스템 속성들과 사용자 속성들의 값을 설정한다.

마지막으로 각 장면 비디오 객체들을 시간 순서대로 정렬한 리스트(Lcv_1)를 포함하는 기본 비디오 객체 (Opv_1)를 생성하고 관련 속성들의 값을 설정함으로써 원시 비디오 데이타에 대한 처리 과정이 완료된다. 생성된 Opv_1은 Lcv_1에 포함된 Ocv_1, Ocv_2, Ocv_3 순서대로 처리되어 결국 사용자에게는 부가적인 정보를 포함한 원시 비디오로 보여지는 것이다.

4.3 파생 비디오 객체 합성

파생 비디오 객체는 다른 비디오 객체들로 구성된 복합 객체로서 기존의 데이타를 이용하여 다양한 내용의 비디오들을 쉽게 표현할 수 있도록 한다.

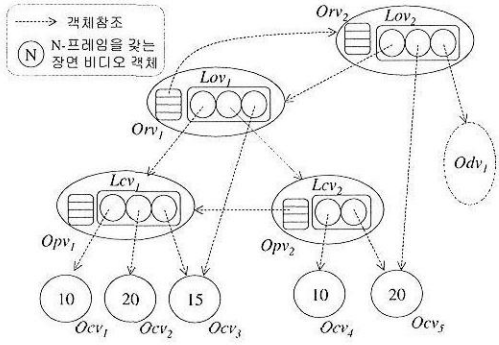

그림 5 파생 비디오 객체 합성

그림 5는 파생 비디오 객체 Orv_1과 Orv_2를 생성하는 예를 보여주는데, 먼저 Orv_1은 Opv_1, Opv_2, 그리고 Ocv_3으로 구성되어 그 순서대로의 내용을 표현한다. 여기서 Ocv_3은 Opv_1에도 포함되므로 결국 Orv_1에서 두번 보여지게 되는데, 이러한 구성은 과거 장면의 회상과 같은 경우에 유용하게 사용될 것이다. 다음으로 Orv_2는 Orv_1, Ocv_5, 그리고 Odv_1로 구성되는데, 결국 그 내용은 Opv_1, Opv_2, Ocv_3, Ocv_5, Odv_1의 순서로 보여질 것이다. 이 때, 동적 비디오 객체 Odv_1은 자주 변하는 내용을 보여줄 때, 예를 들어 그림 4의 객체들 중에서 접근 횟수가 많은 2개를 보여주려는 경우에 유용하게 사용될 수 있다.

한편 비디오 객체에 관련된 여러 정보들을 속성으로 저장할 수 있는데, 그 값으로는 정의1에서와 같이 기본 값 뿐만 아니라 다른 객체의 식별자가 지정될 수 있다. 예를 들어, Opv_1에 비디오 객체 타입의 '다음 비디오'라는 속성이 정의되어 있고 그 값으로 Opv_2의 식별자를 갖는다고 할 때, 이 경우에 Opv_1 객체의 비디오 내용을 보다가 도중에 '다음 비디오' 속성을 선택함으로써 마치 TV 채널을 바꾼 것처럼 Opv_2의 비디오 내용을 볼 수 있다.

4.4 동적 비디오 객체 생성 및 처리

동적 비디오 객체의 생성과 처리는 그림 6의 예를 통해 살펴보자. 먼저 동적 비디오 객체 Odv_1은 Ssv_1이라는 정적 비디오 객체 집합과 Cs_1이라는 요소 결정 조건을 포함한 속성들로 구성되어 있음을 볼 수 있다. 여기서 Ssv_1에 기존의 객체가 삭제되거나 새로운 객체가 추가될 수도 있으며, 그 객체들이 변경되거나 Cs_1 또한 변경 가능하기 때문에 Odv_1이 접근될 때마다 다르게 보여질 수 있는 것이다.

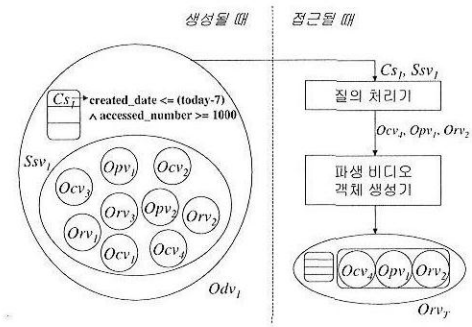

그림 6 동적 비디오 객체의 생성 및 처리

다음으로 동적 비디오 객체의 처리 과정을 살펴보자. Odv_1이 접근될 때 질의처리기가 Ssv_1의 객체들 중 "생성된지 일주일 이내이면서 접근 횟수가 1000회 이상"인 것들, 예를 들어 Ocv_4, Opv_1, 그리고 Orv_2를 차례로 검색해내면 파생 비디오 객체 생성기가 그 객체들을 순서대로 포함하는 임시 파생 비디오 객체(Orv_T)를 생성한다. Orv_T의 속성으로는 Odv_1이 갖고 있는 속성들을 상속받고, 추가로 Odv_1로부터 임시적으로 생성되었다는

정보를 포함하는데, 사용이 끝난 후에 자동으로 삭제된다는 점을 제외하면 일반 파생 비디오 객체와 동일하다.

5. DVID의 성능에 관한 고찰

DVID 모델이 실용적이기 위해서는 무엇보다도 성능에 대한 고려가 필요하다. 먼저 파생 비디오 객체(Orv)는 각기 따로 저장 관리되는 여러 비디오 객체들로 구성되지만 사용자에게는 마치 하나의 비디오처럼 자연스럽게 보여야 한다. 더구나 사용자들이 상호대화식으로 비디오를 볼 수 있기 때문에 비디오 데이타에 대한 빠른 임의 접근 방식까지 제공해야 한다 한가지 해결 방법은 Orv의 재생에 앞서 그것의 모든 요소 객체들을 비중첩화 과정을 거쳐 Ocv들만으로 구성된 비디오로 추상화하는 것이다. 이것은 Orv의 객체 구성 계층의 각 객체를 깊이 우선 방식으로 방문(Depth First Traversal)하면서 각 Ocv에 대해서 그 식별자(Oid)와 포함하는 프레임들의 수(Nfr)를 얻어 Orv내에서의 위치(Ns)와 함께 차례로 저장함으로써, 결국 Orv는 (Ns, Oid, Nfr)로 표현되는 각 Ocv의 논리적 정보들의 리스트로 추상화된다. 만일 동적 비디오 객체를 포함하는 경우에는 그것을 처리하여 임시 파생 비디오 객체를 생성한 후에 추상화 과정을 수행한다. 예를 들면 그림 5의 비디오 객체들은 그림 7과 같이 추상화된다.

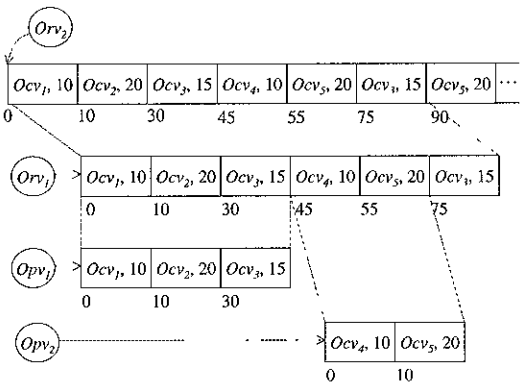

그림 7 비디오 객체의 추상화

다음으로 동적 비디오 객체(Odv)의 명시 방법에 대해서 많은 고려가 요구되는데, 무엇보다도 Ssv를 쉽게 지정하는 방법을 제공하는 것이 필요하다. 여기서는 다음과 같은 네가지의 Ssv 지정 기법을 제안한다

1) Odv에서 특정 객체를 선택하여 자신의 Ssv 내에 포함하도록 지정.
2) 현재 객체에서 특정 Odv를 선택하여 그것의 Ssv 내에 자신이 포함되도록 지정.
3) 각 비디오 객체에 분류 코드를 부여하고, Odv의 Ssv에 특정 분류 코드값을 갖는 모든 비디오 객체들을 포함시키는 방법.
4) Odv는 Ssv 생성을 위한 질의를 포함하고, 전체 비디오 객체를 대상으로 그 질의를 처리한 모든 결과 비디오 객체들을 Ssv 내에 자동으로 포함시키는 방법

임시 파생 비디오 객체(Orv_T)에 대한 효율적 처리 기법 또한 필요하다. 기본적으로 Orv_T는 Odv가 접근될 때 생성되어 대신 사용되다가 더이상 접근이 없을 때 자동으로 삭제된다. 하지만 Odv가 자주 접근된다면 생성된 Orv_T를 따로 저장하여 필요할 때마다 다시 사용하는 것이 훨씬 효율적일 것이다. 이 때 모든 Orv_T를 저장하기보다는 일정 기준에 따라, 예를 들어 특정 시간 동안 접근된 횟수가 일정 기준 이상일 때 저장하는 것이 바람직하며, 아울러 특정 시간 동안 접근된 횟수가 일정 기준 이하일 때 저장된 Orv_T를 삭제하는 것도 필요하다. 한편, 비디오 데이타베이스의 변화로 인해서 저장된 Orv_T가 더이상 올바르지 않을 수 있다. 이런 경우에는 기존의 Orv_T는 삭제되고 Odv가 다시 처리되어 새로운 Orv_T를 생성해야 하는데 이런 경우를 알아내는 연구 또한 필요하다 한가지 방안으로 OODB에서의 스키마 진화에서 사용되는 지연 갱신(deferred update) 방식과 유사하게 객체의 변화를 지정하는 타임스탬프와 그 변화의 승인을 지정하는 타임스탬프를 사용하면 그러한 변화를 알아낼 수 있다.

6. 기존 비디오 데이타 모델에서의 질의 저장 방식과의 비교

기존의 정적 비디오 데이타 모델에서 사용자가 원하는 비디오를 찾기 위해서는 일일이 비디오를 보거나 제목, 키워드 등의 간단한 인덱스를 이용해야 한다. 보다 효율적인 검색을 위해서 질의를 사용하기도 하는데, 사용자가 원하는 비디오를 검색하기 위한 질의를 작성하여 실행시킨 후에 검색 결과로부터 원하는 비디오를 선택하여 감상한다. 만일 특정 질의가 많은 사용자들에 의해 자주 실행된다면 그 질의를 따로 저장했다가 필요할 때마다 재사용하는 것이 보다 효율적일 것이다. 이러한 방식을 질의 저장 방식이라 할 때 본 연구에서의 동적 비디오 객체 방식과는 다음과 같은 차이점들이 있다.

첫째, 질의는 공통된 구조를 갖는 데이타들 중에서 특정 조건을 만족하는 것들을 검색하는데 사용된다 예

를 들어, OODBMS에서의 질의는 특정 클래스의 모든 인스탄스 객체들로부터 원하는 것들을 검색하게 된다. 하지만 DVID의 동적 비디오 객체에서는 사용자가 관심을 갖는 객체들을 따로 집합으로 관리하는데, 그 객체들은 비디오 객체 클래스의 인스탄스들 중에서 일부만 선택되는 것이 일반적이다. 즉, 질의는 공통 구조의 데이타 집합으로부터의 검색인데 비해서 동적 비디오 객체에서는 공통 관심의 데이타 집합으로부터의 검색을 이용하는 것이다.

둘째, 질의의 결과는 임시적인 집합이기 때문에 추가적인 정보를 지속성있게 저장하기 어렵다. 하지만 동적 비디오 객체에서는 주석, 관련 비디오 객체 등 다양한 정보를 지속성있게 저장할 수 있으며, 그 처리 결과로 생성된 임시 정적 비디오 객체는 동적 비디오 객체의 속성들을 상속받으므로 결국 동적 비디오 객체는 결과 집합뿐만 아니라 추가적인 지속적 정보를 제공할 수 있다.

셋째, 질의의 결과는 임시적이기 때문에 다른 객체의 요소로 포함될 수 없다. 이를 가능하도록 하기 위해서는 질의 결과로부터 새로운 지속적 객체를 생성해야 하는데, 이런 방식은 번거러울 뿐만 아니라 질의의 대상이 되는 비디오들의 상태가 변경되었을 때마다 사용자가 질의를 다시 수행시켜 새로운 결과 비디오를 생성해야만 하는 문제점이 있다. 하지만 동적 비디오 객체는 그 자체가 하나의 비디오 객체이기 때문에 다른 비디오의 구성요소로 자연스럽게 사용될 수 있으며, 사용자의 개입없이도 향후 대상 비디오들의 상태 변화를 자동으로 반영할 수 있다.

넷째, 비디오 질의의 검색 결과는 비디오 객체들의 집합이기 때문에 사용자가 원하는 비디오를 선택한 후에야 비로소 일반 비디오 객체와 같은 방식으로 다룰 수 있다. 하지만 동적 비디오 객체는 결과 비디오들을 포함하는 임시 정적 비디오 객체로 처리되므로 결국 기존의 비디오에서와 같은 방식으로 다룰 수 있다.

7. 결론 및 향후 연구

이제까지의 비디오 데이타 모델들은 비디오 데이타를 정적으로 합성하고 사용자에게 보여주는데 주된 관심을 가져왔다. 그러한 정적 비디오 데이타 모델에서는 모든 비디오 편집 작업이 수작업으로 처리되어야 하기 때문에 비디오 데이타의 내용 변화가 빈번한 응용에서 사용되기는 부적합하다.

본 논문에서는 비디오 데이타의 동적 표현에 주된 관심을 갖는 동적 비디오 객체 데이타 모델인 DVID를 제안하였다. 본 연구는 빈번한 비디오 편집 작업이 필요한 VOD, 인터넷 쇼핑, 디지털 도서관 등 여러 비디오 응용들이 있지만 기존의 비디오 데이타 모델에서는 편리한 비디오 편집을 위한 적절한 고려가 없었다는 점에 착안하여 시작되었다. DVID는 객체지향 데이타 모델에 기반한 비디오 모델로서 기존의 비디오 객체들을 이용하여 새로운 비디오 객체를 쉽게 생성할 수 있고 각 객체는 데이타 중복없이 공유될 수 있어서 비디오 편집 작업을 보다 용이하게 지원한다 또한 비디오 편집에서 수작업을 부분적으로 제거하기 위해 시스템이 처리할 수 있는 변화 유형들을 정의하였으며, 그러한 변화를 자동으로 처리하기 위하여 접근될 때 비디오 데이타베이스로부터 명시된 조건을 만족하는 비디오를 동적으로 결정하여 보여주는 동적 비디오 객체를 제안하였다. 결국 동적 비디오 객체에 의해서 비디오 데이타베이스에 발생한 정의된 변화 유형이 자동으로 반영되기 때문에 비디오 편집뿐만 아니라 비디오 관리에 드는 노력을 크게 줄일 수 있게 된다.

향후 연구는 DVID 모델의 실용화에 관심을 둘 계획이다. 먼저 동적 비디오 객체의 성능을 포함한 DVID 모델의 효용성을 평가하여 그것이 대용량의 비디오 데이타와 많은 동시 사용자를 지원하는 환경에서 실제로 사용할 수 있는 정도로 그 성능을 향상시키는 연구가 필요하다. 동적 비디오 객체 대상의 질의 처리도 관심 연구 분야인데, 임시 파생 비디오 객체를 생성한 후에 질의를 처리하거나 동적 비디오 객체 내의 조건들을 질의와 함께 처리하는 방법 등에 관해서 연구할 것이다. 이와 같은 DVID에 관한 보완 연구들에 기반하여 실제 비디오 데이타베이스 응용에 활용하는 동적 비디오 데이타베이스 시스템 DVIDS(Dynamic Video Object Database System)를 개발할 계획이다.

참 고 문 헌

[1] Allen, J.F., "Maintaining knowledge about temporal intervals," Communications of ACM, Vol.26, No.11, pp. 832-842, 1983

[2] Little, T.D.C. and Ghafoor, A., "Interval-based conceptual models for time-dependent multimedia data," IEEE TKDE, Vol.5, No.4, pp. 551-563, 1993.

[3] Weiss, R., Duda, A. and Gifford, D.K., "Composition and search with a video algebra," IEEE Multimedia, Vol.2, No.1, pp. 12-25, 1995.

[4] Oomoto, E. and Tanaka, K., "OVID: Design and implementation of a video-object database system," IEEE TKDE, Vol.5, No.4, pp. 629-643, 1993.

[5] Gibbs, S., Breiteneder, C. and Tsichritzis, D., "Data

modeling of time-based media," in Proceedings of the ACM-SIGMOD, pp. 91-102, 1994.
[6] Hjelsvold, R., "VideoSTAR - A Database for Video Information Sharing," Dr.Ing. Thesis, Norwegian Institute of Technology, 1995.
[7] Hamakawa, R. and Atarashi, A., "Composite Models," The handbook of multimedia information management, pp. 211-236, Prentice Hall, 1997.
[8] Ahanger, G. and Little, T.D.C., "Automatic Composition Techniques for Video Production," IEEE Trans. on Knowledge and Data Engineering, Vol.10, No.6, 1998.
[9] Decleir, C., Hacid, M.S. and Kouloumdjian, J., "Modeling and Querying Video Databases," In Proceedings 24th EUROMICRO'98 Conference Workshop on Multimedia and Telecommunications, 1998.
[10] Khoshafian, S. and Baker, A., "Multimedia and Image Databases," Morgan Kaufmann Publishers, 1996.
[11] Mackay, W.E. and Davenport, G., "Virtual Video Editing in Interactive Multimedia Applications," CACM, Vol.32, No.7, pp. 802-810, 1989.
[12] Ahanger, G., and Little, T.D.C., "A Survey of Technologies for Parsing and Indexing Digital Video," Journal of Visual Communication and Image Representation, Vol.7, No.1, pp. 28-43, 1996.
[13] Patel, N.V. and Sethi, I., "Video Segmentation for Video Data Management," The handbook of multimedia information management, pp. 139-165, Prentice Hall, 1997.

송 용 준
1992년 2월 서울대학교 컴퓨터공학과 졸업. 1994년 2월 서울대학교 컴퓨터공학과 석사. 1994년 3월 ~ 1996년 2월 한국통신 멀티미디어 연구소 근무. 1996년 3월 ~ 현재 서울대학교 컴퓨터공학과 박사과정 재학중. 관심분야는 객체지향 시스템, 멀티미디어 시스템, 데이타베이스

김 형 주
제 26 권 제 1 호(B) 참조