# 구조화된 비디오 문서의 데이타 모델 및 질의어와 색인 기법

류은숙† · 이규철†

## 요 약

비디오 정보는 전자 도서관이나 WWW 및 주문형 비디오(VOD) 시스템과 같은 다양한 응용 분야에서 중요한 요소로 부각되고 있다. 비디오 정보는 특성상 계층적으로 구조화된 문서 형태를 가지기 때문에 본 논문에서는 이를 "구조화된 비디오 문서"라 부른다. 본 논문에서는 구조화된 비디오 문서를 데이타베이스에 효율적으로 저장하고, 검색하기 위한 데이타 모델과 질의어 및 색인 기법을 제안한다. 구조화된 비디오 문서는 논리적인 계층 구조 특성을 지니기 때문에, 본 논문에서는 객체 지향 데이타 모델을 이용하여 비디오 문서를 복합 객체로 모델링하고, 이를 저장하기 위한 객체 타입들을 정의한다. 또한 본 논문에서는 비디오 데이타의 캡션이나 주석을 기반으로 한 내용 기반 검색과 비디오 문서의 논리적 구조를 기반으로 한 구조 기반 검색을 제공하며, 비디오 문서의 시공간 관계 연산을 이용한 검색도 지원한다. 그리고, 구조화된 비디오 문서의 효율적인 질의와 색인 공간의 오버헤드를 줄이기 위해 최적화된 역 색인 기법을 제시한다.

# Data Model, Query Language, and Indexing Scheme for Structured Video Documents

Eunsook Ryu† · Kyuchul Lee†

## ABSTRACT

Video information is an important component of multimedia systems such as Digital Library, World-Wide Web (WWW), and Video-On-Demand (VOD) service system. Video information has hierarchical document structure inherently, so it is named "structured video document" in this paper. This paper proposes a data model, a query language, and an indexing scheme for structured video documents in order to store, retrieve, and share video documents efficiently. In representing structured video documents, the object-oriented data modeling technique is used since the hierarchical structure information can be modeled as complex objects. We also define object types for the structure information. Our query language supports not only content-based retrieval, which means the queries based on the captions or annotations of video data, but also structure-based retrieval, which means the queries based on the structure of video documents, and spatial/temporal relation retrieval for video documents. In order to perform structure queries efficiently, as well as to reduce the storage overhead of indices, an optimized inverted index structure is proposed.

## 1. 서 론

최근 들어 미디어(media)별 처리 기술의 발전, 데이타 입출력 장치와 저장 장치의 발달, 컴퓨터 처리 속도의 향상, 통신 기술의 획기적 발전 등으로 컴퓨터를 이용한 멀티미디어(multimedia)의 실현 환경이 구축되고 있으며, 또한 새로운 다양한 미디어 서비스 요구가 증가되고 있다. 특히 OIS(Office Information System), 디지털 도서관(Digital Library), WWW (World-Wide-Web), VOD(Video On Demand) 등

본 연구는 한국 과학재단 핵심연구(과제번호 : 971-0902-014-2)과제로 수행되었음.

† 충남대학교 공과대학 컴퓨터공학과

멀티미디어 정보 처리를 요구하는 다양한 응용 분야의 출현으로 대량의 멀티미디어 정보를 효율적으로 저장하고, 처리하며, 검색할 수 있는 기능이 요구되고 있다. 특히 비디오 정보는 방송, 교육, 출판과 같은 응용 분야에서 점점 중요한 요소로 부각되고 있다.

비디오 정보에 대한 초기 연구는 비디오 데이타를 물리적인 세그먼트(segment), 즉 연속적인 프레임(frame) 주기로 분할하는 것이었으나, 최근에는 논리적인 세그먼트 단위로 구조화하려는 방법들이 연구[1,2,3,4,5,6,7,8,9,10]되고 있다. 예를 들면, 비디오 데이타의 내용 자체를 비디오 분석 기술을 이용해 분석하기보다는, 비디오 데이타를 논리적인 세그먼트로 구조화한 후, 여기에 텍스트나 자연어 형태로 된 설명문(description), 즉 주석(annotation)을 부착해서 이 설명문을 검색하는 방식이 많이 사용되고 있는 것이다. 이러한 방식을 비디오 데이타의 내용 기반(content-based) 검색, 혹은 주석 기반(annotation-based) 검색이라 하는데, 이때 검색 효율을 높이기 위해 내용 기반 색인(indexing) 방법도 함께 연구되고 있다.

하지만 대부분의 기존 연구들은 비디오 데이타의 구조 정보에 대한 데이타 모델과 질의어(query language) 및 효율적인 색인 방법들을 충분히 제공하지 못하고 있다. 예를 들면, 오늘 방송된 9시 뉴스 프로그램에서 스포츠 소식 장면(scene)과 관련된 스포츠 경기 프로그램을 찾고자 할 때, 기존의 물리적인 색인 방법만으로는 이러한 질의를 검색할 수 없다. 즉, 이와 같은 질의에 대한 결과를 얻기 위해서는 우선적으로 비디오 데이타에 대한 프로그램이나 장면과 같은 계층 구조 정보가 있어야 하며, 또한 비디오의 각 장면을 공통의 의미적인(semantic) 내용끼리 분류해 놓아야 한다. 위의 질의 예라면, 스포츠 경기 종목별로 장면을 분류하거나, 스포츠 뉴스의 화제별로 장면을 분류할 수 있다. 이러한 각 장면의 특성은 장면에 달린 캡션(caption)이나 주석 문으로부터 추출될 수 있으며, 이를 내용 기반 검색의 키워드(keyword)로 이용할 수 있다. 이러한 키워드에 대한 색인들은 질의의 검색 효율을 높이기 위해 적절한 색인 구조를 가지고 제공되어야 한다.

따라서 비디오 데이타를 효율적으로 저장 및 공유하고, 검색하기 위해서는 데이타베이스를 기반으로 한 새로운 데이타 모델과 다양한 질의 연산을 제공해야 하며, 또한 비디오 데이타의 효율적인 질의를 위해 색인 기능이 제공되어야 한다.

본 논문에서는 구조화된 형태로 데이타베이스에 저장하거나 검색할 수 있는 비디오 데이타를 구조화된 비디오 문서(structured video document)라 부르고, 이에 대한 데이타 모델과 질의어 및 내용 기반 색인 방법들을 제공한다. 구조화된 비디오 문서는 계층적인 구조로 표현될 수 있기 때문에, 본 논문에서는 비디오 문서를 객체 지향(object-oriented) 기술을 기반으로 한 복합 객체(complex object)로 모델링하고, 기본적인 질의뿐만 아니라 비디오 문서의 내용을 기반으로 한 질의, 그리고 구조화된 정보를 기반으로 한 질의 및 비디오 문서의 시공간 관계성 질의 등 다양한 질의 연산을 제공하며, 특히 구조화된 비디오 문서의 질의를 위해 효율적인 색인 기능을 제공한다.

본 논문의 구성은 다음과 같다. 2장에서는 비디오 문서를 데이타베이스에 효율적으로 저장하고, 검색하기 위해서 고려해야 할 모델링 요건들을 기술하고, 3장에서는 이를 토대로 한 비디오 문서의 모델링 방안을 제안한다. 4장에서는 비디오 문서에 대한 다섯 가지 형태의 질의 표현식에 대해 설명하고, 5장에서는 비디오 문서의 색인 구조에 대해 기술한다. 6장에서는 비디오 문서의 모델링에 대한 기존의 연구 방법들을 비교해서 살펴보고, 마지막으로 본 논문의 향후 연구 방안에 대해 7장에서 설명한다.

## 2. 비디오 문서 모델링의 요건 분석

비디오 문서를 데이타베이스에 저장하고 관리할 때, 이 들간에 정보의 손실이 없도록 충분히 모델링되어야 한다. 본 장에서는 비디오 문서를 데이타베이스에 효율적으로 저장하고, 관리 및 검색하기 위해 고려해야 할 모델링 요건들을 기술한다.

### 2.1 비디오 문서의 구조

비디오 정보 자체는 원시(raw) 데이타이지만 대부분의 경우 비디오 문서 전체를 저장하고, 검색하는 대신 비디오의 각 장면을 분할해서 저장하고 원하는 형태의 장면만을 검색할 수 있는 기능[4]을 요구하고 있다. 비디오 문서의 분할은 비디오 내에 있는 원하는 대상 물체를 비디오 배경이나 다른 물체로부터 구분 시켜 의미를 가지는 형태로 만드는 작업으로서

해당되는 대상 물체에는 캡션이나 주석을 부가하여, 원래의 비디오를 체계적으로 구성시킨다. 이와 같이 구조화된 비디오 문서는 이를 데이타베이스에 저장하거나 검색할 때, 그리고 비디오 문서를 색인하는 과정에서 시스템 효율에 영향을 미친다.

비디오 문서를 의미 있는 장면들로 분할해서 구조화하는 방법은 크게 세 가지로 나눌 수 있다. 첫째는 여러 가지 비디오 분석 기법들을 사용하여 비디오의 의미를 추출하는 방법이다. 예를 들면, 카메라의 이동 방향이나, 물체의 명암, 색상, 이동 방향 등으로 장면을 추출할 수 있다. 그러나 이런 경우 비디오가 담고 있는 일반적인 의미를 추출하는 것이 매우 어려우므로 일반적인 비디오 검색 시스템에서 사용하기에는 문제가 있다. 둘째는 사람이 비디오를 미리 보고 인위적으로 의미 있는 장면들에 주석을 달아 이를 추출하는 방법이다. 이런 경우 사람이 인식할 수 있는 모든 내용을 주석으로 처리할 수 있기 때문에 일반적인 비디오 검색 시스템으로 확장이 가능하다. 하지만 사용자의 관점에 따라 다양한 주석이 표현될 수 있으므로 일관성을 잃기 쉬우며, 비디오의 의미를 사용자가 일일이 부여하여야 한다는 단점[7]도 있다. 세 번째는 위에서 설명한 두 가지 방법을 동시에 취하는 것이다. 이런 경우에는 비디오 문서가 다양한 계층으로 나뉘어 관리되거나, 각 계층간의 연관 관계가 복잡해 질 수 있다. 비디오 문서를 데이타베이스에 효율적으로 저장하고, 검색하기 위해서는 비디오 문서를 의미 있는 단위로 구조화시키는 작업이 기본적으로 요구된다.

### 2.2 비디오 문서의 검색

데이타베이스에 저장된 비디오 문서에서 특정 비디오를 검색하고자 할 때 기존의 데이타베이스 질의어만으로는 효과적인 검색을 기대할 수 없다. 예를 들어, TV 뉴스 프로그램을 모두 검색하는 질의는 비디오 문서의 제목만으로도 검색이 가능하지만, 뉴스 프로그램 중 '대학 입시 개정안'에 대한 기사 장면만을 검색하고자 할 때는 비디오의 내용을 추출해서 내용 기반 검색을 해야 한다. 또한 9시 뉴스의 세 번째 기사를 검색하는 질의는 내용 기반으로도 검색이 어려운 예로서, 이런 경우에는 뉴스 비디오의 논리적인 구조 정보를 기반으로 한 검색이 요구된다. 따라서 비디오 문서를 효율적으로 검색하기 위해서는 다양한 형태의 질의가 지원되어야 한다.

### 2.3 비디오 문서의 색인

비디오 문서의 색인은 비디오 문서의 검색을 더 빠르고 효율적으로 지원하기 위해 제공된다. 비디오 색인은 크게 다음의 두 가지 형태로 구분 지울 수 있다. 즉, 비디오로부터 움직임이나 색상과 같은 하위 레벨의 측정치를 추출해내어 비디오 순서를 위한 키(key)로 사용하는 방법이 있으며, 또 한가지 방법은 비디오 색인을 위해 미리 정의된 색인 항목(term)들의 집합들을 사용하는 것으로써, 이러한 항목들을 이용해 시간적인 스트림에 따라 비디오를 첨부시키거나 주석을 붙이는 방법이 있다. 색인 과정으로는 사용자가 수동으로 키워드와 그 내용을 기재하는 수동식 색인과 이미지나 비디오 분석 방법을 이용한 자동식 색인, 그리고 이 두 가지 과정을 절충한 반자동식 색인이 있다. 기존의 비디오 색인 방법으로는 비디오에 대한 기술 내용 위주로 비디오를 계층화(stratification)시킨 후 색인 하는 방법[1]과 이미 색인된 개별 색인들을 대수(algebraic) 연산을 이용해 결합시키고, 각 계층간의 연관 관계를 제공함으로써 의미적인 계층 구조 정보를 색인하는 방법[6]이 제안되었다. 결론적으로, 비디오 문서의 색인은 데이타베이스에 저장된 비디오 문서의 구조와 밀접한 관련을 가지며, 비디오 문서 구조에 가장 적합한 비디오 색인 구조가 설계되어야 한다.

### 2.4 비디오 문서의 내용 특성 추출

비디오 문서를 모델링하기 위해서는 비디오 문서가 내포하고 있는 내용의 특성들을 분류할 필요가 있다. 비디오 문서를 이용한 내용 기반 검색이나 주석 기반 검색 시스템을 설계하고자 할 때, 이러한 분류[11]는 비디오 문서의 구조화는 물론 검색 및 색인 구조에도 많은 영향을 미칠 수 있기 때문이다. 내용 특징 중에는 비디오 문서로부터 직접 사용 가능하지 않은, 내용에 독립적인 특성과 내용에 의존적인 특성이 있다. 예를 들면, 비디오의 제작비와 같은 것은 내용에 독립적인 특성인 반면, 줄거리는 비디오를 보아야 알 수 있다. 또한 가장 많이 나타나는 색상이나 물체의 위치와 같이 시간 간격 내에서 단 하나의 프레임에 기초하여 비디오를 서술할 수 있는 특성이

있는가 하면, 여러 시간 간격의 흐름에 근거하여 서술해야 하는 특성이 있다. 그밖에 도메인 모델에서 그 특성의 성질에 따라 명칭을 부여할 수 있는 특성이 있고, 도메인에 독립적인 모델 특성이 있다. 예를 들면, 농구 경기에서 농구공의 추적과 같은 특성은 후자의 경우에 해당하고, 농구공의 패스, 드리블, 덩크 슛 등은 전자의 특성에 해당된다. 이외에도 공통의 의미적인(semantic) 내용에 따라 그 특성을 분류할 수도 있다.

### 2.5 비디오 문서의 시간 정보

비디오 문서는 이미지 데이타와 달리 시간 개념을 포함하고 있다. 사용자의 질의가 비디오의 시간적인 정보에 의존하는 경우에는 이러한 시간 정보를 필요로 한다[2,12]. 따라서 비디오 문서 모델에서 이러한 시간 데이타를 지원할 수 있어야 할뿐만 아니라 시간 간격간의 연관 관계를 유지할 수 있어야 한다.

### 2.6 비디오 문서의 공유

비디오 문서는 데이타베이스에 저장될 때뿐만 아니라, 저장된 후에도 의미 있는 장면에 설명적인 데이타를 추가하여 새로운 관계를 생성할 수 있으며, 한 의미 있는 장면은 종종 여러 다른 장면에 포함되거나 겹칠 수 있다. 특히, 비디오 문서가 대량의 데이타인 점을 고려하면, 서로 포함 관계에 있는 장면과 설명 자료는 서로 공유해서 사용하는 것이 바람직하다. 따라서, 비디오 문서의 구조 정보를 공유할 수 있는 저장 구조 모델이 요구된다.

## 3. 비디오 문서의 구조 정보 모델링

본 장에서는 2 장에서 기술한 비디오 문서 모델링의 요건들을 토대로 비디오 문서를 효율적으로 관리하고, 검색할 수 있는 비디오 문서의 논리적 계층 구조 특성을 설명하고, 이러한 구조 정보를 위한 타입들을 정의한다. 또한 각 계층 구조에 효율적으로 접근할 수 있는 메커니즘과 비디오 문서의 저장 구조 모델을 제안한다.

### 3.1 비디오 문서의 논리적 계층 구조 특성

비디오 문서는 의미 있는 논리적 단위로 분할됨으로써, 각 구조간에 계층적 특성을 지닌 문서 구조로 표현될 수 있다. 그림 1은 비디오 문서의 논리적 계층 구조를 나타낸 것으로, 객체 지향 개념의 복합 객체로 표현한 것이다.

본 논문에서는 비디오 문서의 논리적인 구조를 모델링하기 위해 비디오 문서의 응용 범위(domain)를 TV 방송으로 가정하였다. 그림 1에서 보는 바와 같이 하나의 비디오 문서는 여러 프로그램(program)들로 구성되고, 한 프로그램은 여러 개의 씬(scene)들로 구성되며, 한 씬은 다시 하나 이상의 샷(shot)으로 구성되어 있음을 알 수 있다.

예를 들면, TV 방송에서 광고에 이어, 뉴스와 스포츠가 각각 순서대로 방송된다고 했을 때, 광고, 뉴스, 스포츠 등은 프로그램에 해당한다. 또한 광고 프로그램은 여러 회사의 광고를 모아서 한꺼번에 방영하는데, 이 때 각 회사의 광고를 하나의 씬으로 구성할 수 있으며, 마찬가지로 뉴스 프로그램에서 각 뉴스 기사와 스포츠 소식, 일기 예보 등을 각각 하나의 씬으로 구성할 수 있다. 일기 예보를 전하는 씬에서는 기상 캐스터와 구름 사진 등을 각각 샷으로 구성할 수 있다.

샷은 연속적인 프레임(frame)들로 구성되는데, 여기서 프레임은 비디오 데이타의 물리적인 단위에 해당하며 샷, 씬, 프로그램 등은 본 논문에서 사용하는 비디오 문서의 논리적인 단위이다. 특히 씬 객체는 사용자가 관심 있어하는 샷들을 모아 재구성될 수 있으며, 하나의 샷이 여러 씬 객체에 중복해서 포함될 수 있다. 예를 들면, 똑같은 광고가 여러 번 방송될 수 있기 때문에 그림 1에서처럼 첫 번째 광고 프로그램의 B_scene은 그 다음 광고 프로그램에서 또다시 방송될 수 있다. 또한 스포츠 뉴스에서는 뒤에 방송될 스포츠의 중요 장면을 미리 자료 화면으로 방송할 수 있다. 그림 1에서 Shot44가 이와 같은 예이다.

그림 1의 논리적 계층 구조에서 실제 데이타는 단말 노드, 즉 샷에만 저장되며, 중간 노드들은 논리적 구조를 나타내는데 사용된다. 또한 자식과 부모 노드 사이에는 is-part-of 관계성이 존재하게 된다. 각 노드에 붙여진 번호는 각 노드를 유일하게 구별하기 위한 번호이며, 이에 대한 설명은 3.3절에서 언급될 것이다.

비디오 문서에서 각 장면의 특성들은 장면에 달린 캡션이나 주석문에 있는 텍스트로부터 추출되거나

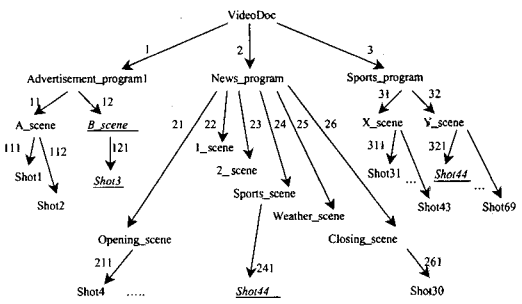


그림 1. 비디오 문서의 논리적 계층 구조

비디오 분석 툴을 이용하여 비디오 프레임으로부터 추출될 수 있으며 이를 내용 기반 검색의 키워드로 이용할 수 있음을 가정한다. 그림 1과 같은 논리적 계층 구조의 각 객체를 정의하면 다음과 같다.

***정의 1*** : 비디오 문서 객체, $V=(vid, Pt_i\ A_i)$ 여기서,
- $vid$ : 비디오 문서 객체의 고유 식별자를 나타낸다.
- $Pt_i$ : 프로그램 객체 $P(Pt_i \in P)$의 집합을 나타내며, $t_i(1 \le i \le n)$는 객체의 순서를 의미한다.
- $A_i(1 \le i \le n)$ : 비디오 문서 객체에 대한 특성들을 나타낸다.

***정의 2*** : 프로그램 객체, $P=(pid, Ct_i\ A_i)$ 여기서,
- $pid$ : 프로그램 객체의 고유 식별자를 나타낸다.
- $Ct_i$ : 씬 객체 $C(Ct_i \in C)$의 집합을 나타내며, $t_i(1 \le i \le n)$는 객체의 순서를 의미한다.
- $A_i(1 \le i \le n)$ : 프로그램 객체에 대한 특성들을 나타낸다.

***정의 3*** : 씬 객체, $C=(cid, Ot_i\ A_i)$ 여기서,
- $cid$ : 씬 객체의 고유 식별자를 나타낸다.
- $Ot_i$ : 샷 객체 $O(Ot_i \in O)$의 집합을 나타내며, $t_i(1 \le i \le n)$는 객체의 순서를 의미한다.
- $A_i(1 \le i \le n)$ : 씬 객체에 대한 특성들을 나타낸다.

***정의 4*** : 샷 객체, $O=(oid, F[i, j], A_i)$ 여기서,
- $oid$ : 샷 객체의 고유 식별자를 나타낸다.
- $F[i, j]$ : 비디오 데이타의 연속적인 프레임들을 나타내며, $[i,j](1 \le i, j \le n)$는 객체의 순서를 의미한다.
- $A_i(1 \le i \le n)$ : 샷 객체에 대한 특성들을 나타낸다.

### 3.2 비디오 문서의 구조 정보를 위한 타입 정의

본 논문에서 제시한 비디오 문서의 논리적 구조는 복잡한 계층성을 갖는 특성을 지니고 있기 때문에, 본 논문에서는 객체 지향 데이타 모델을 이용하여 비디오 문서의 계층 구조 정보를 모델링하고, 이를 저장하기 위한 타입들을 정의한다. 그림 2는 비디오 문서의 계층 구조 정보를 지원하기 위한 타입 계층 구조이다.

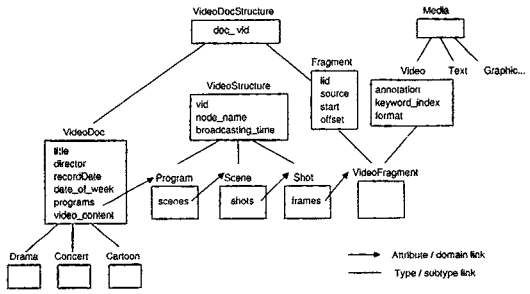


그림 2. 비디오 문서의 구조 정보를 지원하기 위한 타입 계층

비디오 문서 구조의 루트 타입은 VideoDocStructure이며, 이의 서브 타입으로 VideoDoc, Fragment 타입을 갖는다. VideoDoc 타입 밑에는 다양한 종류의 비디오 문서 형태가 올 수 있는데, 여기서처럼 Drama, Concert, Cartoon 등이 올 수 있다. 이러한 비디오 문서는 VideoDoc 타입의 모든 속성(property)을 계승(inherit) 받을 수 있으며, 각 문서의 속성에 맞게 새로운 속성을 정의할 수 있다. 비디오 구조의 모든 구성 객체는 각각 Program, Scene, Shot이라는 타입으로 정의되며, VideoStructure 타입의 서브 타입으로 정의된다.

그림 2의 화살표는 애트리뷰트와 도메인 링크를 나타내고 있는데, VideoDoc 타입의 애트리뷰트인 programs는 도메인으로 Program 타입을 가지며, 또한 Scene과 Shot 타입도 각각 scenes와 shots 애트리뷰트의 도메인으로 정의된다

Video 타입의 상위 타입은 Media이며, 이는 Video 타입뿐만 아니라 텍스트나 이미지, 그래픽과 같은 멀티미디어 데이타에 대한 상위 타입으로도 정의될 수 있다. Video 타입과 Fragment 타입의 서브 타입은 VideoFragment이며, 동시에 frames 애트리뷰트의 도메인으로 정의된다. VideoFragment 타입은 비디오 문서의 저장 구조를 위해 제공되며, 3.4절에서 설명될 것이다.

그림 2의 타입 구조에서 VideoDocStructure, VideoStructure, Fragment, Media 타입은 추상 타입(abstract type)으로 선언된다. 이러한 타입은 서브 타입들의 공통된 특성을 추출하여 기술하기 위한 타입으로 사용된다.

그림 2에 대한 각 타입들의 정의를 데이타베이스 표준 언어인 SQL3[13,14]의 구문에 맞추어 정의하면 그림 3과 같다. SQL3은 객체 지향 모델의 지원이 가능하도록 현재의 관계 데이타 표준인 SQL2를 확장하여 현재 세계 표준으로 진행중인 언어로서, 관계 데이타베이스 시스템과 객체 지향 시스템의 장점을 모두 수용하도록 표준 작업이 진행되고 있다.

```
CREATE OBJECT TYPE VideoDocStructure
(PROTECTED      doc_vidString,
FUNCTION        ancestor(e1 String, e2 String)
                RETURNS String
FUNCTION        children(e1 String, e2 String)
                RETURNS LIST(String)
FUNCTION        sibling(e1 String)
                RETURNS LIST(String));

CREATE OBJECT TYPE VideoDoc
        UNDER VideoDocStructure
(title          Text,
 director       Text,
 recordDate     Date,
 date_of_week   String,
 programs       LIST(Program),
 video_content  Video,
 FUNCTION       GetVideoDoc (v1 Video)
                RETURNS Video);

CREATE OBJECT TYPE VideoStructure
(PROTECTED      vid             String,
        node_name               String,
        broadcasting_time       Time );

CREATE OBJECT TYPE Program
        UNDER VideoStructure
(scenes LIST(Scene) );

CREATE OBJECT TYPE Scene
        UNDER VideoStructure
(shots LIST(Shot) );

CREATE OBJECT TYPE Shot
        UNDER VideoStructure
(frames LIST(VideoFragment) );
```

그림 3. 비디오 문서의 구조 정보에 대한 타입 정의

먼저 VideoDocStructure에 대한 타입 정의에서 doc_vid는 비디오 문서를 유일하게 구별하게 하는 키 애트리뷰트이며, VideoDoc 타입 정의문에서는 비디오 문서에 일반적으로 필요한 제목이나, 책임자, 녹화 날짜, 요일 등에 관한 정보를 기술하고 있다. programs 애트리뷰트는 도메인으로 Program 타입의 LIST를 갖는데, 이는 애트리뷰트의 값으로 순서성이 있는 여러 값을 가짐을 의미한다. video_content 애트리뷰트는 비디오 문서에 나오는 모든 비디오 데이타의 전체 내용을 일괄하여 저장하고 있는 애트리뷰트로서 그 비디오 데이타에 대한 포인터를 갖는다. 한편, VideoDoc 타입에서 제공하는 GetVideoDoc 메소드(method)는 비디오 문서 전체 내용을 한꺼번에 보고자 할 때 사용될 수 있다. VideoDocStructure 타입 정의에서 제공하는 3개의 메소드는 3.3절에서 설명된다.

VideoStructure 타입 정의문에서는 비디오 문서의 논리적 계층 구조상의 각 노드를 유일하게 식별할 수 있는 키 애트리뷰트 vid와 노드명, 방송 시간 등을 정의하고 있다. 이 타입은 Program, Scene, Shot 타입에서 공통으로 사용되는 애트리뷰트들을 추출하여 정의한 추상 타입이다.

Media에 대한 타입 정의(그림 4)에서 start와 end는 비디오 문서가 프리젠테이션될 때의 시작 시간과 끝나는 시간을 나타내는 애트리뷰트이며, 이 정보는 비디오 문서들간의 시간 관계성을 파악하는데 이용된다. position 애트리뷰트는 프리젠테이션 되는 위치 정보를 나타낸다. 본 논문에서 비디오 문서의 실제 내용은 content 애트리뷰트에 Blob(Binary large object)으로 저장된다. Blob은 정형화되지 않은 이진 스트림(binary stream)이며, 현재 대부분의 상용 관계 또는 객체 지향 DBMS는 멀티미디어 데이타 저장을 위해 Blob을 지원하고 있다.

그림 4의 Media 타입에서 제공하는 메소드들은 두 미디어간의 시간 관계성을 나타내기 위해 제공된다. 이 시간 관계성은 Allen[15]이 제안한 13개의 시간 관계성 중 실제로 필요한 7개만을 제공한다. 그 이유는 나머지 관계성은 equal을 제외한 나머지 여섯 개 관계성의 역(inverse)이기 때문에, 연산시 피연산 객체의 순서를 바꾸면 되기 때문이다. 예를 들어, 'before(a, b)'는 'after(b, a)'와 동일하기 때문이다. 위의 메소드에서 equal은 두 미디어의 프리젠테이션 시작

```
CREATE OBJECT TYPE Media
(PUBLIC      start      Time,
             end        Time,
PRIVATE      content    Blob,
             position   Point
FUNCTION  equal (m1 Media, m2 Media)
    RETURNS BOOLEAN
FUNCTION  before (m1 Media, m2 Media)
    RETURNS BOOLEAN
FUNCTION  meets (m1 Media, m2 Media)
    RETURNS BOOLEAN
FUNCTION  during (m1 Media, m2 Media)
    RETURNS BOOLEAN
FUNCTION  overlaps (m1 Media, m2 Media)
    RETURNS BOOLEAN
FUNCTION  starts (m1 Media, m2 Media)
    RETURNS BOOLEAN
FUNCTION  finishes (m1 Media, m2 Media)
    RETURNS BOOLEAN);
```

그림 4. Media 타입 정의

시간과 끝나는 시간이 같은지, before는 m2 전에 m1이 프리젠테이션 되는지, meets는 두 미디어가 이어서 프리젠테이션 되는지를 검사한다. 그리고, 한 미디어가 다른 미디어의 프리젠테이션 시간에 포함되는지를 나타내는 during, 두 미디어가 일부 겹쳐서 프리젠테이션 되는지를 나타내는 overlaps, 두 미디어의 프리젠테이션 시작 시간이나 끝나는 시간이 같은지를 각각 나타내는 starts와 finishes 메소드가 있다. 이러한 메소드들은 비디오 데이타와 그 밖의 각 미디어 데이타 타입으로 계승되어 사용된다.

Video에 대한 타입 정의는 그림 5와 같다.

그림 5에서 format 애트리뷰트는 비디오 객체의 포맷을 기술하기 위해 사용되며, keyword_index 애트리뷰트는 비디오의 내용 기반 색인을 지원하기 위해 제공된다. annotation 애트리뷰트는 비디오의 내용 검색을 지원하기 위해 붙여진 설명 데이타로서, 이를 이용해 contains 메소드는 주어진 키워드가 비디오 설명에 포함되어 있는지를 검색하는 내용 기반 검색 연산을 정의한다. 나머지 연산들은 비디오에 관련된 고유 연산들인데, 크게 세 가지 부류로 분류할 수 있다. 즉, 두 개의 비디오 클립을 합성하는 overlay, 비디오 클립을 여러 상태로 변화시키는 filter, 비디오의 일 초당 프레임 개수를 수정하는 frameSize, 비디오의 색상을 변환시키는 colorTransform 연산 등

```
CREATE OBJECT TYPE Video
                UNDER Media
(PRIVATE annotation       Text,
         keyword_index    InvertedIndex,
         format           String,
FUNCTION contains( v1 Video,
   search_expr varchar(max_pattern_length))
                RETURNS BOOLEAN
FUNCTION  overlay( v1 Video, v2 Video )
                RETURNS Video
FUNCTION  filter(v1 Video, a Attri)
                RETURNS Video
FUNCTION  frameSize(v1 Video, i Integer)
                RETURNS Video
FUNCTION  colorTransform(v1 Video, c Color)
                RETURNS Video
FUNCTION  extract( v1 Video, a Attri )
                RETURNS Video
FUNCTION  equal( v1 Video, v2 Video )
                RETURNS BOOLEAN
FUNCTION  fadeIn( v1 Video )
                RETURNS Video
FUNCTION  fadeOut( v1 Video )
                RETURNS Video
FUNCTION  morphing( v1 Video, v2 Video )
                RETURNS Video
FUNCTION  ghosting( v1 Video )
                RETURNS Video
FUNCTION  replicate(v1 Video )
                RETURNS Video
FUNCTION  playDuration( v1 Video, t Time)
                RETURNS Video
FUNCTION  forward( v1 Video, o Offset )
                RETURNS Video
FUNCTION  backward( v1 Video, o Offset )
                RETURNS Video
FUNCTION  position( v1 Video, t Time,
                a area)   RETURNS Video);
```

그림 5. Video 타입 정의

은 속성 변환 연산에 속하며, 비디오 클립에서 특정 색상의 마스크만 뽑아내는 extract, 두 비디오가 같은가를 체크하는 equal 연산 등은 속성 추출 연산에 속한다. 또한, 영상이 서서히 나타나거나 사라지는 fadeIn과 fadeOut, 한 영상에서 다른 영상으로 서서히 변환되는 morphing, 영상에 잔상을 만들어 주는 ghosting, 전체 영상을 여러 개의 화면 형태로 나누어 보여 주는 replicate, 지정한 시간 동안만 동작시키는 playDuration, 특정한 부분으로 시작점을 이동시켜 다시 작동시키는 forward와 backward, 지정한

시간동안 특정 영역의 비디오 객체를 보여주는 position 연산 등은 프리젠테이션 연산에 해당한다.

### 3.3 계층 구조 접근 메커니즘

비디오 문서의 각 구조 정보를 효과적으로 검색하기 위해서는 비디오 문서의 계층 구조를 효과적으로 표현하는 것이 필요하다. 본 논문에서는 구조 정보 접근을 위해 HID (Hierarchical Identifier)를 제공한다. 그림 1에 붙여진 번호는 바로 이 HID인데, 루트 노드로부터 자신의 노드까지 이르는 경로에 있는 각 노드들의 HID를 접합(concatenate)시킴으로 해서 만들어지며, 처음으로 HID가 부여된 노드 계층으로부터 하위 레벨로 내려갈수록 레벨수도 증가한다. 본 논문에서는 계층 구조의 각 노드 타입이 이미 결정되어 있으므로 각 노드 타입의 레벨을 미리 알 수 있다. 예를 들면, 그림 1에서와 같이 프로그램 노드의 레벨이 1이면, 씬과 샷은 각각 2와 3이 되며, 프로그램 노드의 레벨이 2라면, 씬과 샷도 이에 따라 각각 3과 4가 될 것이다.

본 논문에서는 이러한 HID를 이용하여 비디오 문서의 계층 구조에 대한 접근 방법으로 3개의 함수, 즉 *ancestor, children, sibling* 함수를 제공한다. *ancestor* 함수는 어떤 노드로부터 사용자가 지정한 노드 타입의 조상 노드를 구하는 함수이며, *children* 함수는 어떤 노드로부터 사용자가 지정한 노드 타입의 후손 노드를 구하는 함수이다. 마찬가지로, *sibling* 함수는 어떤 노드의 형제 노드들을 구하는 함수이다. 따라서 이와 같은 함수를 이용하면, 원하는 계층 구조의 노드를 추가 정보의 접근 없이 간단한 계산으로 즉시 접근할 수 있다. 각 함수에 대한 정의는 다음과 같다.

***정의 5*** : *ancestor(node, t)*
= *shift_right(HID(node), k)* 여기서,
- *node : 계층 구조상에 있는 임의의 노드*
- *t : 노드 타입*
- *k : HID(node)의 길이 - t의 레벨 수*
- *shift_right : 주어진 노드의 HID를 k만큼 오른쪽으로 shift*

***정의 6*** : *children(node, t)*
= *shift_left(HID(node), k)* 여기서,
- *node : 계층 구조상에 있는 임의의 노드*
- *t : 노드 타입*
- *k : t의 레벨 수 - HID(node)의 길이*
- *shift_left : 주어진 노드의 HID를 k만큼 왼쪽으로 shift*

***정의 7*** : *sibling(node)*
= *shift_left(shift_right(HID (node), 1), 1)*
여기서,
- *node : 계층 구조상에 있는 임의의 노드*
- *shift_right : 주어진 노드의 HID를 1만큼 오른쪽으로 shift*
- *shift_left : shift_right한 노드의 HID를 1만큼 왼쪽으로 shift*

*ancestor* 함수를 이용하여 원하는 조상 노드를 구하는 예는 다음과 같다. 그림 1에서 News_program.Opening_scene.Shot4의 프로그램 조상 노드는 HID '211'을 2 만큼 오른쪽으로 shift 시켜 얻은 News_program이 된다.

*ancestor(News_program.Opening_scene.Shot4, Program)*
*= shift_right( HID(News_program.Opening_scene. Shot4), 3-1)*
*= shift_right ( '211', 2)*
*= '2' (=HID(News_program))*

마찬가지로, 그림 1에서 A_scene의 샷 후손 노드는 HID '11'을 1 만큼 왼쪽으로 shift 시켜 얻을 수 있는데, '11x'를 갖는 2개의 노드들이 그 답이 된다.

*children(Advertisement_program.A_scene, Shot)*
*=shift_left(HID(Advertisement_program.A_scene), 3-2)*
*=shift_left ( '11', 1)*
*='111', '112'*
*(=HID(Advertisement_program.A_scene.Shot1),*
*HID(Advertisement_program.A_scene.Shot2))*

또한 A_scene의 형제 노드에 대한 값은 다음과 같이 얻어질 수 있다.

*sibling(Advertisement_program.A_scene)*
*=shift_left(shift_right(HID(Advertisement_program.A_scene), 1), 1)*
*=shift_left(shift_right('11', 1), 1)*
*=shift_left('1', 1)*
*='11', '12'*

*따라서, 형제 노드는 '12' (=HID(Advertisement_program.B_scene))가 된다.*

*ancestor*와 *children, sibling* 함수는 타입 VideoDocStructure에 메소드로 제공될 수 있다(그림 3). 일반적으로 객체 지향 데이타베이스에서는 이를 구현하기 위해 복합 객체 상에 부모-자식 양방향 링크(link)를 설정하여 사용하지만, 이 때 조상 또는 손자 노드들을 접근하기 위해서는 그 사이에 나오는 모든 노드들을 접근해야 하므로 디스크 접근 시간이 많이 걸린다. 그 반면에 본 논문의 HID 방법은 디스크 접근 없이 간단한 계산만으로 복합 객체상의 어떤 노드라도 자유롭게 접근 가능한 장점을 가지고 있다.

### 3.4 비디오 문서의 저장 구조

그림 1의 비디오 문서 계층 구조에서 실제 데이타를 가지고 있어야 하는 노드들은 shot 단말 노드들이다. 하지만, 이 단말 노드들이 실제 데이타를 가지도록 하면 여러 가지 문제가 야기된다.

첫째는 각 구조의 단말 노드들이 각각 데이타를 가지면 많은 중복이 발생할 수 있다. 예를 들면, 그림 1에서 HID '241'을 가진 News_program.Sports_scene.Shot44와 HID가 '321'인 Sports_program.Y_scene.Shot44는 한 구조 내에서 서로 다른 단말 노드이지만 실제로 같은 비디오 데이타이다. 둘째는 비디오 문서 전체 정보를 한꺼번에 보고자 할 때, 데이타를 각 단말 노드들이 각각 가지고 있으면, 이를 일일이 모두 검색, 접근하여 프리젠테이션 해야 하므로 성능의 심각한 저하를 가져 올 수 있다.

따라서 위와 같은 문제점을 해결하기 위하여 본 논문에서는 각 구조의 단말 노드들이 실제 데이타를 갖는 대신, 이들 노드들의 데이타를 모아서 한꺼번에 blob 형태로 저장하여 같은 내용의 데이타에 대한 공유가 가능하게 하고, 각 단말 노드는 이에 대한 포인터 'source'와 source에서의 자신의 시작 위치와 길이 정보를 갖는 offset 정보를 가지도록 하였다. 이를 구현해 주는 타입이 Fragment이며, Fragment는 위의 source와 offset, 샷 객체의 HID인 lid를 애트리뷰트로 갖도록 정의되었다. 실제 비디오 문서는 압축된 형태로 저장되기 때문에 압축된 비디오에 대한 시작 위치와 길이 정보를 가지고 비디오 문서를 검색하려면, 이러한 정보만으로도 원하는 프레임을 꺼낼 수 있는 새로운 메소드가 제공되어야 한다. 이러한 메소드는 VideoFragment 타입에 정의될 수 있다. 하지만, 이와 같은 메소드는 또다른 응용 프로그램 범주에 속하므로 본 논문에서는 언급하지 않는다. 다음은 Fragment에 대한 타입 정의문이다.

```
CREATE OBJECT TYPE Fragment
  UNDER VideoDocStructure
  (lid        String,
   source     Media,
   start      Number,
   offset     Number);
```

그림 2에서 Video 타입과 Fragment 타입의 공통 서브 타입으로 VideoFragment 타입을 볼 수 있다. 이 VideoFragment 타입에서 자신이 속한 비디오 문서의 doc_vid는 VideoDocStructure 타입으로부터 계승받아 사용된다. 예를 들면, 그림 1의 비디오 문서를 TV_video_doc이라고 할 때, VideoFragment의 객체들은 그림 6과 같다. VideoFragment11과 VideoFragment12 두 객체는 서로 다른 lid를 가지고 있지만, start와 offset을 통해 서로 같은 객체임을 알 수 있다. 이와 같은 방법을 통해 우리는 데이타의 중복을 없애고, 각 구조 사이에 데이타를 공유할 수 있으며, 문서 전체 정보의 프리젠테이션 성능을 향상시킬 수 있는 장점을 얻을 수 있다.

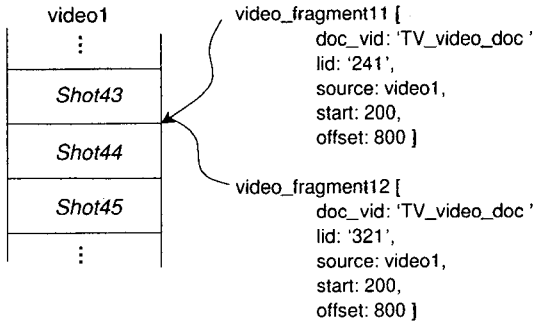


그림 6. 비디오 데이타의 공유

## 4. 비디오 문서의 질의 표현식

본 논문에서는 비디오 문서의 효과적인 검색을 위하여 5가지 형태의 질의를 지원하는 비디오 질의 표현식을 확장하였다. 5가지 형태의 질의는 다음과 같다.

첫째는 애트리뷰트 질의로서 title이나 recordDate와 같은 일반 애트리뷰트의 값을 질의할 수 있는 것이며, 둘째는 내용 기반 질의로서 비디오 문서의 내용을 기반으로 한 질의를 의미한다. 셋째는 구조 기반 질의로서 비디오 문서의 논리적인 계층 구조에서 각 구조 정보에 대한 질의를 의미한다. 넷째는 두 비디오 문서 구성 요소간의 시간 및 공간 관계성 질의로서 Media 타입 정의에서 제공하는 시간 관계 연산자와 Video 타입 정의에서 제공하는 공간 관계 연산자를 이용한 질의를 의미하며, 마지막 유형은 비디오 고유 연산자 질의로서 그림 5의 Video 타입 정의에서 제공하는 고유 연산자를 이용한 질의를 의미한다.

물론 위의 5가지 형태의 질의를 복합하여 좀 더 강력한 질의를 만들 수 있다. 본 장에서는 먼저 확장된 질의 표현식을 기술하고, 이러한 확장된 질의 표현식을 이용한 5가지 형태의 질의어를 제시한다. 질의 언어의 구문은 SQL3 표준에 따라 작성되었다.

### 4.1 확장된 질의 표현식

본 논문에서는 비디오 문서의 구조를 지원하기 위해 [16]의 연구에서 확장 제안한 질의 표현식 중에서 두 가지를 사용한다. 확장된 질의 표현식은 상하위 어느 레벨의 노드를 접근할 수 있도록 하는 component 연산과 노드 리스트의 각 개별 노드를 접근할 수 있는 LIST 연산이다. 이들 연산들은 메소드 형태로 구현될 수 있는데, Fragment 타입의 메소드로 등록되어, 하위 타입들이 이를 계승받아 사용할 수 있다. 각 연산에 대한 정의는 다음과 같다.

***정의 8*** : *연산자 component(parameter)는 다음과 같이 정의된다.*

- *component : 비디오 문서의 계층 구조에서 자신의 조상이나 후손 노드들을 자유롭게 접근할 수 있는 연산자*
- *parameter : 자신이 접근하고 싶은 레벨을 지정*

***정의 9*** : *연산자 L[i] 또는 L[i:j]는 다음과 같이 정의된다.*

- *L : 노드 이름이거나 component 연산이 될 수 있다. 노드들의 리스트에서 특정 노드들을 접근하기 위한 연산자.*
- *i, j : 임의의 정수이거나 '$'가 될 수 있다. 예를 들어, L[i]는 i번째 노드를 추출하며, L[$]일 경우는 마지막 노드를 추출한다. L[i:j]인 경우는 i부터 j번째까지의 서브 리스트를 추출한다.*

*component* 연산은 3.3절에서 설명된 ancestor와 children 함수를 이용하여 구현될 수 있으며, component 연산자의 파라미터인 *parameter*는 다음의 표 1과 같이 정의된다. 표 1의 파라미터를 적용하여 그림 1에 대해 component 연산과 LIST 연산을 수행하면 그 결과는 다음과 같다.

- Advertisement_program1..component(+)
  = {A_scene, B_scene, Shot1, Shot2, Shot3}
- Weather_scene..component(/)
  = {Weather_scene, News_program}
- Sports_program..component($)
  = {Shot31, ... Shot69}
- News_program..component(>)
  = {Sports_program}
- Advertisement_program1..component(2)
  = {Shot1, Shot2, Shot3}
- News_program[2]..scene[$] : 두 번째 뉴스 프로그램의 마지막 씬
- Shot[1 : 3] : 첫 번째, 두 번째, 세 번째 샷들
- News_program..component(>)[1] : 뉴스 프로그램 다음에 이어서 방송되는 프로그램

표 1. 질의 표현식에 사용되는 파라미터의 종류와 의미

| 파라미터 | 의 미 |
|---|---|
| * (+) | 0(1) 또는 그 이상의 하위 모든 레벨의 노드들 |
| / (−) | 0(1) 또는 그 이상의 상위 모든 레벨의 노드들 |
| 0 | 자기 자신 노드 |
| $ | 자신 밑에 있는 모든 단말 노드들 |
| ^ | 자신의 루트 노드 |
| > | 자신과 같은 레벨의 노드중 자신 노드 다음에 있는 형제 노드들 |
| < | 자신과 같은 레벨의 노드중 자신 노드 전에 있는 형제 노드들 |
| <> | 자신과 같은 레벨의 모든 형제 노드들 |
| i (-i) | 자신으로부터 i 레벨 밑(위)의 노드들 |
| i~j (-i~-j) | 자신으로부터 i부터 j 레벨 사이 밑(위)의 노드들 |

### 4.2 비디오 문서의 애트리뷰트 질의

다음 질의는 일반 애트리뷰트의 값을 질의하는 전형적인 질의문 예이다. 예를 들어, 비디오 문서 중에서 doc_vid가 'TV_pro'인 문서의 title과 recordDate를 질의하면 다음과 같다.

```
SELECT   v..title, v..recordDate
FROM     VideoDoc v
WHERE    v..doc_vid = 'TV_pro'
```

여기서 v는 VideoDoc을 대표하는 객체 변수(object variable)이며, 도트를 두번 쓰는 것은 해당 메소드를 호출하는 SQL3의 구문 형식이다

### 4.3 비디오 문서의 내용 기반 질의

본 논문에서는 비디오 문서의 질의를 위해 구조화된 비디오 문서에 텍스트 형태로 된 주석을 부착해 이 주석을 검색하는 방식을 사용한다. 이러한 비디오 문서의 내용 기반(content-based) 질의는 비디오 문서의 논리적인 구조 정보와 함께 제공됨으로써, 사용자에게 더 다양한 질의를 제공할 수 있다.

다음 질의 예는 비디오 문서의 뉴스 프로그램 중에서 '날씨'와 '비'라는 키워드를 포함하는 모든 씬들을 검색하는 질의이다.

```
SELECT   p..scenes
FROM     VidoeDoc v, v..programs p
WHERE    v..doc_vid = 'TV_pro' AND
         p..node_name = 'News_program'
         AND contains(p..scenes, '날씨' AND
         '비')
```

위 조건 절의 contains는 그림 5의 Video 타입에 정의된 내용 기반 질의를 지원하는 메소드이며, 비디오 문서의 각 장면에 달린 주석에서 이와 같은 키워드를 검색하게 된다.

### 4.4 비디오 문서의 구조 기반 질의

본 논문에서는 비디오 문서의 논리적 구조에 대한 질의도 가능하다. 예를 들어, "9시 뉴스의 3번째 기사를 검색하라"라는 질의는 비디오 문서에 대해 흔히 할 수 있는 질의인데, 본 논문에서 제안한 논리적 계층 구조와 LIST 연산을 이용하여 다음과 같이 기술할 수 있다.

```
SELECT   p..scenes[3]
FROM     VideoDoc v, v..programs p
WHERE    v..doc_vid = 'TV_pro' AND
         p..node_name = 'News_program'
         AND p..broadcasting_time
           = '21:00:00'
```

다음은 뉴스 프로그램 다음에 방송되는 프로그램을 검색하는 질의 예이다. 여기서 p..component(>)[1]은 뉴스 프로그램과 같은 레벨의 논리적인 구조 노드 중에서 다음에 이어지는 형제 노드 중 첫 번째를 나타낸다. 일반 텍스트 문서와 달리 비디오 문서는 데이타 자체에 연속적인 시간의 의미를 내포하고 있기 때문에 LIST 연산자가 유용하게 쓰일 수 있다.

```
SELECT   p..component(>)[1]
FROM     VideoDoc v, v..programs p
WHERE    v..doc_vid = 'TV_pro' AND
         p..node_name = 'News_program'
```

다음은 더 복잡한 질의 예로서, "뉴스 기사 중에서 키워드가 '축구'인 장면과 똑같은 장면을 포함하고 있는 스포츠 프로그램을 검색하라"라는 질의를 기술한 것이다. 여기서 'component($)'의 의미는 대상 노드의 모든 단말 노드를 의미한다.

```
SELECT   q
FROM     VideoDoc v, v..programs p q,
         (p.. scenes..component($)) s
WHERE    v..doc_vid = 'TV_pro' AND
         p..node_name = 'News_program'
           AND
         q..node_name = 'Sports_program'
           AND
         contains(p..scenes, '축구') AND
         s IN q..scenes..component($)
```

### 4.5 비디오 문서의 시간 및 공간 관계성 질의

본 논문에서는 두 비디오 문서 구성 요소간의 시간 관계성을 나타낼 수 있는 질의를 제공한다. 예를 들어, 앞절에서 설명한 "뉴스 프로그램 다음에 방송

되는 프로그램을 검색"하는 질의 예를 시간 관계성 질의로 표현할 수 있다. 그림 4의 Media 타입에 정의된 메소드 'meets'를 이용하면 다음과 같이 나타낼 수 있다. 여기서, meets 연산은 두 미디어가 이어서 프리젠테이션 되는지를 검사하는 연산자이다. 이외에도 equal, before, during, overlaps, starts, finishes 등과 같은 연산자를 이용하여 시간 관계성 질의를 표현할 수 있다.

```
SELECT   q
FROM     VideoDoc v, v..programs p q
WHERE    v..doc_vid = 'TV_pro' AND
         p..node_name = 'News_program'
         AND meets(p, q)
```

비디오 문서의 공간 관계성 질의는 비디오 문서의 프리젠테이션과 관련이 있다. 비디오 데이타는 시간 속성을 가진 동적 데이타이므로, 비디오 데이타에서 표현할 수 있는 구성 요소간의 공간 관계성도 시간 속성을 가진다. 예를 들어, 뉴스 프로그램 화면에서 오른쪽 상단 부분의 일정 영역에 기사 내용과 관련된 자료 화면을 제공한다고 할 때, 이 영역만을 검색하려면 그림 5의 Video 타입에 정의된 메소드 'position'을 이용할 수 있다. 다음 질의 예에서 position 연산은 지정한 시간동안('10'초)에 특정 영역(x_origin, y_origin, width, height)의 비디오 객체를 프리젠테이션하는 연산자이다.

```
SELECT   position(s, 10, (20,30,5,5))
FROM     VideoDoc v, v..programs p,
         p..scenes [3] s
WHERE    v..doc_vid = 'TV_pro' AND
         p..node_name = 'News_program'
```

### 4.6 비디오 고유 연산자 질의

본 논문에서는 비디오 고유 연산자 질의를 제공한다. 다음 질의는 "뉴스 프로그램의 세 번째 장면과 똑같은 장면을 포함하고 있는 스포츠 프로그램을 검색"하는 질의를 기술한 것이다. 여기서 equal 연산자는 두 비디오가 같은가를 검사하는 것으로서, Video 타입의 메소드로 정의된 비디오 고유 연산자이다.

```
SELECT   q
FROM     VideoDoc v, v..programs p q
WHERE    v..doc_vid = 'TV_pro' AND
         p..node_name = 'News_program'
           AND
         q..node_name = 'Sports_program'
           AND
         equal(p..scenes[3], q..scenes)
```

이와 같이 본 논문에서는 비디오 문서에 대한 내용을 기반으로 다양한 질의를 만들 수 있을 뿐만 아니라 구조 정보를 기반으로 한 질의도 지원할 수 있다. 또한, 비디오 문서의 시간 및 공간 관계성 질의와 비디오 고유 연산자 질의도 지원할 수 있다. 본 논문에서는 이러한 정보를 복합적으로 이용하여서도 질의를 할 수 있다.

## 5. 비디오 문서의 색인 구조

본 장에서는 구조 질의를 효율적으로 처리하면서도 색인이 차지하는 저장 오버헤드(storage overhead)를 상당히 줄일 수 있는 비디오 문서의 색인 구조에 대해 설명한다.

본 논문에서는 비디오 문서에 대한 내용 기반 검색을 위해 Video 타입 정의문에서 contains라는 메소드와 annotation 및 keyword_index 애트리뷰트를 제공한다. annotation 애트리뷰트의 값으로부터 추출된 키워드는 색인 구조에 이용되는데, 본 논문에서는 이러한 색인 구조를 keyword_index 애트리뷰트의 도메인인 InvertedIndex 타입에서 제공한다. 이러한 내용 기반 검색은 정보 검색 시스템의 텍스트 문서에 대한 내용 기반 검색과 같다고 볼 수 있다. 대부분의 정보 검색 시스템에서는 전문(full-text)을 효율적으로 접근하기 위하여 역 색인(inverted index) 방법을 사용하고 있다. 본 논문에서도 비디오 문서의 색인 구조를 지원하기 위해 역 색인 방법을 사용한다.

따라서 역-리스트(inverted-list)에는 해당 항목(term)이 나타나는 모든 비디오 문서의 식별자(doc_vid)와 비디오 문서의 계층 구조상에 나타나는 각 노드 식별자, 즉 vid를 저장한다. 또한 구조 질의를 효율적으로 처리할 수 있도록 직접 요소 접근(direct element access)을 제공하기 위해서는, 각 노드들의 모든 색인 항목들을 역 색인에 포함시켜야 한다.

본 논문에서 제안한 비디오 문서의 복합 객체 계층 구조에서는 중간 노드들이 노드들간에 구조적인

관계만을 나타내고, 데이타에 대한 정보는 단말 노드가 가지고 있다. 그리고 각 단말 노드의 색인 항목들은 해당 비디오 문서의 주석으로부터 추출된다. 하지만 이와 같은 구조에서 중간 노드들이 직접 문서 데이타를 가지고 있지 않더라도 하위 트리에 나타난 데이타들을 모두 중간 노드들의 데이타로서 고려해야 한다. 예를 들면, 그림 7과 같이 각 중간 노드들이 하위 트리에 나타난 모든 색인 항목들을 가지고 있어야 한다. 여기서 그림 7을 자세히 살펴보면, 각 중간 노드들은 그들의 자식 노드들이 가지고 있는 색인 항목 중에서 공통된 색인 항목을 가지고 있다는 것을 알 수 있다. 따라서 모든 노드의 공통 색인 항목은 '동계 올림픽'이다. 그림 7에 도시된 색인 항목의 역-리스트는 다음과 같이 구성된다.

inverted-list (동계올림픽) =
{video1, News_program, Sports_program,
News_program.Scene3, News_program.Scene4,
Sports_program.Scene1, Sports_program.Scene2,
News_program.Scene4.Shot20,
News_program.Scene4.Shot47,
Sports_program.Scene2.Shot47,
Sports_program.Scene2.Shot50}
inverted-list (결승전) = {Sports_program.Scene2,
Sports_program.Scene2.Shot47,
Sports_program.Scene2.Shot50}
inverted-list (우승자 인터뷰) =
{News_program.Scene4.Shot20}

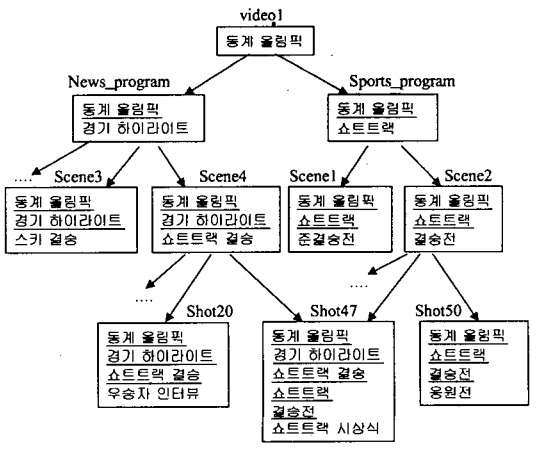


그림 7. 중복된 색인 항목을 가진 비디오 문서의 논리적 계층 구조

앞의 역-리스트에서 보듯이, 각 중간 노드들이 그들의 자식 노드들로부터 공통된 색인 항목을 가지고 있다는 사실로부터, 본 논문에서는 구조화된 비디오 문서에 대하여 최적화된 역 색인 구조를 구성할 수 있다[17]. 즉, 자식 노드의 공통된 색인 항목들은 그들의 부모 노드만 가지도록 하고, 자식 노드에서는 이를 제거함으로써 중복을 피할 수 있는 것이다. 이러한 중복된 색인 항목을 제거하면 색인 항목에 대한 역-리스트의 크기가 감소될 수 있다. 이와 같은 최적화된 역 색인과 3.3절에서 설명한 ancestor와 children 함수를 이용하면 훨씬 적은 색인 공간을 가지고서도 비디오 문서 구조 내의 어떤 요소들도 쉽게 접근할 수 있다. 그림 8은 최적화된 색인 항목을 가진 논리적 계층 구조를 나타낸다. 따라서, 그림 8에 도시된 최적화 색인의 역-리스트는 다음과 같이 간단하게 구성된다.

inverted-list (동계올림픽)= {video1}
inverted-list(결승전) = {Sports_program.Scene2}
inverted-list (우승자 인터뷰)
= {News_program.Scene4.Shot20}

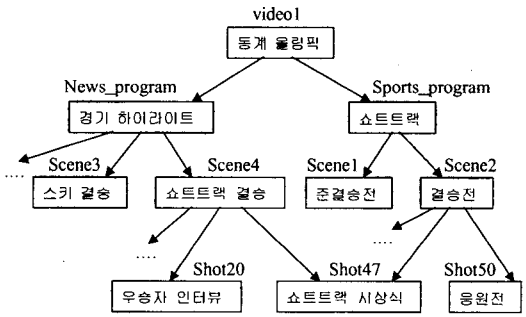


그림 8. 최적화된 색인 항목을 가진 비디오 문서의 논리적 계층 구조

일반적인 역-리스트에 들어가는 식별자 수는 색인 항목이 나타나는 문서 요소의 수에 따라 기하급수적으로 증가하여 점점 더 커진다. 하지만, 본 논문과 같이 색인 구조를 구성한다면, 자식 노드에서 부모 노드로 최적화시킬 수 있는 키워드의 비율이 크면 클수록 색인 크기는 감소한다. 하지만, 기존의 색인 방법에서는 공통되는 키워드가 있더라도 역-리스트에 있는 식별자 수를 감소시키는데 별 도움이 되지 않으

므로 색인 크기는 변하지 않는다.

다음은 색인 구조를 위한 타입을 보여주고 있다. InvertedIndex 타입은 항목과 그 항목이 위치한 포스팅(posting) 정보를 애트리뷰트로 갖는다. Posting 클래스는 비디오 문서 식별자(doc_vid)와 계층 구조상의 노드 식별자(vid)들로 이루어진다.

```
CREATE OBJECT TYPE InvertedIndex
  (   term          String,
      posings       SET(Posting) );
CREATE OBJECT TYPE Posting
  (   doc_vid       String,
      vid           String );
```

## 6. 관련연구

최근 들어, 비디오 문서가 점차 보편화되고, 초고속 통신망이 확산되면서 비디오 문서에 관한 연구가 많은 관심의 대상이 되고 있다. 본 장에서는 비디오 문서의 모델링과 질의어 및 색인 구조 관점에서 기존의 연구들을 비교하였다.

Hjelsvold의 연구[2]에서는 비디오 문서의 구조를 모델링하고, 비디오 스트림에 달린 주석을 색인한다. 또한 비디오 문서의 검색 결과가 중복되었을 때 중복된 객체들간의 시간적 관계 및 내용 기반 검색을 표현할 수 있는 확장된 질의어를 제공한다. 하지만, 비디오 문서의 구조를 기반으로 한 질의를 고려하지 않았다.

Jiang[3]은 VideoText라는 시스템에서 비디오 세그먼트의 내용을 기반으로 한 데이타 모델링과 확장된 질의어를 제공한다. 특히, 질의 결과에 가중치를 주기 위해서 데이타베이스와 정보 검색 시스템을 통합한 비디오 데이타베이스 시스템을 제공한다. 이 연구에서는 시간 관계 연산자 등 다양한 질의 표현식을 제공하고 있으나, 구조 기반 질의를 지원하지 않으며, 색인에 대한 고려가 없다.

Oomoto[4]는 OVID라는 시스템에서 비디오 객체를 모델링하고, 비디오의 주석을 기반으로 한 검색 방법을 제안하고 있다. 비디오 객체의 서술 데이타에 대한 일반화(generalization) 계층 구조를 제공함으로써, 서술 데이타를 공유할 수 있는 메커니즘을 제공한다. 또한 비디오 객체를 재구성할 수 있는 다양한 연산과 질의 기능도 제공하고 있다. 하지만 비디오 객체의 클래스 계층 구조와 이에 대한 모델을 지원하지 않기 때문에 각 비디오 객체간의 구조적인 관계나 이를 이용한 질의를 제공하지 못하며, 비디오 데이타의 색인에 대한 고려도 언급되지 않았다.

Zhang[5]의 연구는 이미지 분석 기법과 비디오 스트림을 shot의 집합으로 분할하는 기법을 이용한 세그멘테이션 기반 모델로서, 뉴스 비디오 프로그램을 도메인으로 사용하여 각 이미지 프레임의 공간적 구조와 전체 프로그램의 시간적 구조를 기반으로 한 모델을 제공한다. 또한 내용 기반 비디오 색인 방법을 제안하였다. 하지만 비디오 데이타에 대한 체계적인 구조가 없으며 본 논문에서와 같은 다양한 질의 표현식을 제공하지 못한다.

Weiss[6]는 그의 연구에서 비디오 대수(algebra)를 제안하고 내용 기반 검색을 지원하는 대수적 비디오 모델과 비디오 데이타의 생성, 시간적 관계, 공간적 배치 구조, 설명 기술 등 비디오 대수 연산을 제공한다. 또한 중첩 논리 구조(nested logical structure)를 이용하여 비디오 데이타를 추상화한다. 하지만 이 연구에서는 질의어에 대한 대안을 제공하지 못하고 있다.

[7]의 연구에서는 내용 기반 검색 및 주석 기반 검색을 통합하는 비디오 데이타 모델을 제안하고 있다. 검색에 사용되는 비디오 데이타의 내용이나 주석을 물리적 데이타로부터 의미 데이타로 추상화하는 과정에서 이 중간 데이타 계층을 모델링하고, 각 계층에 독립적인 질의 유형을 제시하였다. 하지만 각 계층 구조간의 효율적인 접근 메커니즘을 제공하지 못하고 있으며, 표준화된 SQL 질의 표현식과 색인에 대한 고려가 이루어지지 않았다.

[8]의 연구에서는 비디오 데이타의 계층적인 구조를 기반으로 한 메타데이타(metadata) 모델을 제안하고 있다. 특히, 이 연구에서는 비디오 데이타의 특성을 7가지 관점에서 추출하고, 메타데이타를 크게 내용 기반 정보와 내용 독립 정보로 분류한 후, 각각을 더 세분화해서 다양하게 분류하고 있다. 이 연구에서 제안된 메타데이타 모델은 다양하게 분류된 메타데이타와 비디오 데이타의 논리적인 계층 구조를 지원할 수 있다. 또한, 비디오 데이타에 대한 다양한 검색 연산과 시공간 연산을 제공하고 있다. 하지만, 비디오 데이타와 메타데이타 모델에 대한 질의 표현

식과 색인 구조에 대한 언급이 없으며, 각 계층 구조간의 효율적인 접근 메커니즘을 제공하지 못하고 있다.

비디오 주석 시스템 VIRON[9]에서는 단일한 주석 시스템 구조를 속성 스키마, 개념적 주석 구조, 비디오 인덱스 스트림 구조로 분할하여 주석을 이용한 비디오 데이타 질의 시스템을 모델링 하였다. 특히, 비디오 객체 내에 포함되어 있는 내용 데이타와 시간 데이타를 분리하여 이들을 각각 속성 스키마와 비디오 인덱스 스트림에 저장 관리하고, 이를 이용한 시간 관계 질의 및 의미 관계 질의를 제공한다. 이 시스템에서는 속성 스키마의 데이타 구조가 ISA 관계로 구성되기 때문에 비디오 객체의 클래스 계층 구조를 지원하지 않는다. 따라서, 비디오 객체간의 구조적인 관계나 이를 이용한 질의를 제공하지 못한다.

QBIC[18]은 IBM에서 구현한 내용 기반 이미지 및 비디오 시스템이다. 이 시스템에서는 컴퓨터가 인식하기 쉬운 속성들 즉, 색이나 질감 등을 내용으로 저장하고 있기 때문에 사용자에게 사건이나 상황묘사와 같은 의미의 검색을 제공하지 못한다. 또한 비디오 데이타에 대한 구조적인 모델링도 미약하며, 질의 표현식도 다양하지 못하다.

위에서 살펴본 바와 같이 내용 기반 비디오 문서의 모델링과 검색에 관한 연구가 많이 이루어지고 있으며, 비디오 문서의 도메인에 따라 모델링이 다양해 질 수 있다. 표 2에서는 기존 연구와 본 연구를 데이타 모델과 질의 언어 및 색인 구조 관점에서 비교 설명하고 있다.

대부분의 연구에서 비디오 문서를 계층 구조적인 객체로 모델링하고 있으나, 본 논문에서와 같이 각 계층 구조간의 효율적인 접근 방법과 같은 구체적인 메커니즘을 제시하지 못하고 있는 실정이다. 또한, 본 논문에서는 비디오 타입에 대한 정의를 SQL3를 이용하여 정의하고 있는데, 타입 정의문내에서는 내용 기반 연산자와 다양한 비디오 고유 연산자를 메소드로 제공한다.

질의 언어 측면을 살펴보면, 대부분의 기존 연구에서는 비디오 데이타에 대한 내용 기반 검색을 제공하고 있기 때문에, 내용 기반 질의 연산자와 SQL을 이용한 질의 언어를 제공하고 있다. 이외에 최근의 연구에서는 시공간 관계 연산 질의도 함께 제공하고 있는 추세이다. 하지만, 기존의 연구에서는 비디오 문서의 논리적 구조를 기반으로 한 다양한 연산과

표 2. 기존 연구와의 비교

| 구 분 | 기 존 연 구 | 본 논문의 연구 |
|---|---|---|
| 데이터 모델 | ▸비디오 데이타 자체에 대한 논리적인 계층 구조 정보 모델링 [2,3,4,5,6, 7,8,9] | ▸비디오 데이타 자체에 대한 논리적인 계층 구조 정보 모델링<br>▸각 계층 구조간의 효율적인 접근 방법 제공<br>▸비디오 데이타 타입 정의<br>▸비디오 데이타의 고유 연산자 추출 |
| 질의 언어 | 내용 기반 질의 외에,<br>▸시간 관계성 질의 [2,3,8,9] 및<br>▸공간 관계성 질의 [8]를 지원<br>▸SQL 표준을 기반으로 한 질의어 제공[4,9] | ▸내용 기반 질의,<br>▸논리적 구조 기반 질의,<br>▸시간 관계성 질의,<br>▸공간 관계성 질의,<br>▸비디오 고유 연산자 질의를 지원<br>▸SQL3 표준을 기반으로 함 |
| 색인 구조 | ▸내용 기반 색인 구조[4,5,6,9] | ▸내용 기반 색인 구조 (최적화된 역 색인 구조) |

검색을 제공하지는 못하고 있다. 본 연구는 기존의 연구에서 제공하는 질의를 모두 제공하고 있으며, 이외에 비디오 데이타의 논리적인 구조를 기반으로 한 질의를 제공한다. 또한 본 논문에서는 다양한 비디오 고유 연산자를 제공함으로써, 비디오 고유 연산자 질의도 지원할 수 있다.

비디오 데이타의 내용 기반 검색을 제공하는 대부분의 연구에서는 내용 기반 색인 구조를 지원하거나 데이타 특성에 맞는 물리적인 색인 구조를 지원하고 있다. 이 중에서 내용 기반 색인 구조를 지원하는 연구들은 대부분이 정보 검색 시스템에서 사용하는 색인 구조를 지원하고 있다. 본 논문에서도 비디오 데이타에 대한 내용 기반 색인 방법을 채택하고 있으나, 기존의 연구와 달리 본 논문에서는 색인이 차지하는 저장 공간을 줄이면서 비디오 데이타의 논리적인 구조 정보 질의를 효율적으로 처리할 수 있는 최적화된 역 색인 구조를 제안하고 있다.

그밖에, 비디오 데이타의 저장 구조를 위해 본 논문에서는 비디오 데이타의 특성뿐만 아니라, 비디오 문서의 논리적 구조 특성을 고려한 저장 구조 모델을 제안하고 있다. 즉, 본 논문에서는 비디오 데이타의 중복을 없애고, 문서 전체 정보의 프리젠테이션 성능

을 향상시킬 수 있는 효율적인 저장 구조를 제공한다.

## 7. 결론

비디오 문서의 논리적인 계층 구조 특성은 비정형적인 비디오 문서에서 사용자가 원하는 요소를 추출하는데 매우 유용하다. 본 논문에서는 이와 같은 특성을 지원하기 위해 비디오 문서의 데이타 모델과 질의 언어, 그리고 색인 구조를 설계하였다. 비디오 문서의 데이타 모델에서는 객체 지향 데이타 모델을 이용하여 비디오 문서의 계층 구조 정보를 모델링하고, 이를 저장하기 위한 객체 타입들을 정의하였다. 또한 비디오 문서의 계층 구조를 효과적으로 접근하기 위한 메커니즘을 제공하고, 비디오 데이타의 공유 방법에 대해 기술하였다. 본 논문에서는 다양한 질의 연산을 제공하고 있는데, 기본적인 질의뿐만 아니라 비디오 문서의 내용을 기반으로 한 질의와 구조화된 정보를 기반으로 한 질의, 시공간 관계성 질의, 비디오 고유 연산자 질의 등을 제공하고 있다. 또한 구조화된 비디오 문서의 효율적인 질의와 색인 공간을 위해 최적화된 역 색인 구조를 제공한다.

한편, 비디오 문서는 시간 및 공간적인 특성을 가지고 있기 때문에, 비디오 문서의 검색 결과에 대한 신속하고 효율적인 프리젠테이션 기능이 요구된다. 또한 사용자의 질의를 효율적으로 수행하기 위해 본 논문에서 제공하는 연산자 외에도 더 다양한 질의 연산자들이 지원되어야 한다. 예를 들어, 쇼트트랙 천미터 경기에서 1위로 우승한 선수의 프로필 비디오를 검색할 때, 각 경기 장면과 경기에 참가한 선수의 프로필 비디오가 서로 하이퍼링크로 연결되어 있으면 더 효율적으로 검색할 수 있으므로 이를 지원하는 질의가 제공되어야 한다. 그밖에 텍스트, 이미지, 그래픽, 오디오, 비디오 등 다양한 미디어 데이타가 혼합된 멀티미디어 문서에서 각 미디어의 구조 정보 모델링을 유지하면서 이를 하나의 통합된 구조로 모델링하는 방안도 고려할 수 있다. 이러한 추가적인 요구들은 본 논문의 추후 연구라 할 수 있다.

## 참 고 문 헌

[1] Thomas G. Aguierre Smith and Glorianna Davenport, "The Stratification System: A Design Environment for Random Access Video," *In Workshop on Networking and Operating System Support for Digital Audio and Video*, pp. 250-261, 1992.

[2] Rune Hjelsvold and Roger Midtstraum, "Modelling and Querying Video Data," *Proc. of the 20th VLDB Conference*, pp. 686-694, 1994.

[3] Haitao Jiang, Danilo Montesi, Ahmed K. Elmagarmid, "VideoText Database Systems," *International Conference on Multimedia Computing and Systems,* pp. 344-351, 1997.

[4] Eitetsu Oomoto and Katsumi Tanaka, "OVID: Design and Implementation of a Video-Object Database System," *IEEE Transactions on Knoweldge and Data Engineering*, Vol.5, No.4, pp. 629-643, 1993.

[5] HongJiang Zhang, et al., "Automatic parsing and indexing of news video," *IEEE Multimedia Systems,* pp. 256-266, 1995.

[6] Ron Weiss, Andrzej Duda, "Composition and Search with a Video Algebra," *IEEE MultiMedia*, Vol.2, No.1, pp. 12-25, 1995.

[7] 김기병, 김형주, 내용 기반 검색 및 주석 기반 검색을 통합하는 비디오 데이타 모델의 설계 및 구현, 정보과학회논문지(C), 3권, 2호, pp. 115-126, 1997.

[8] Park, Y., Yongkeol Kim, Seongil Jin and Wan Choi, "Hierarchical Structure-based Metadata Model for Video Database Application," *the ISCA 13th international conference Computer And Their Applcations'98*, pp. 242-245, 1998.

[9] 김기욱, 김형주, "비디오 주석 시스템의 설계 및 구현", 정보과학회논문지(B), 24권, 6호, pp. 588-597, 1997.

[10] 류은숙, 구조화된 멀티미디어 문서 모델링에 관한 연구, 충남대 대학원 박사학위 논문, 1998.

[11] Ramesh Jain and Arun Hampapur, "Metadata in Video Databases," *SIGMOD RECORD,* Vol. 23, No.4, pp. 27-33, 1994.

[12] T. D. C. Little, G. Ahanger, et at., "Interval-based conceptual model for time-dependent

multimedia data," *IEEE Transactions on Knoweldge and Data Engineering*, Vol.5, No.4, 1993.

[13] ISO/IEC JTC1/SC 21/WG3 *DBL SQL3*, 1994.

[14] L. Gallagher, "SQL Generic ADT pakages," *ISO/ IEC JTC1/SC21/WG3 DBL*. 1991.

[15] J. F. Allen, "Maintaining Knowledge about Temporal Intervals," *Comm. of the ACM*, Vol.26, No 11, pp. 823-843, 1983.

[16] Eunsook Ryu, Kyuchul Lee, "A Multimedia Query Language for Handling Multi-Structure Information," *Proc. of the Fifth International Conference on Database Systems for Advanced Applications*, pp. 333-342, 1997.

[17] Kyuchul Lee, Y. K. Lee and P. Bruce Berra, "Management of Multi-structured Hypermedia Documents: A Data Model, Query Language, and Indexing Scheme," *Multimedia Tools and Applications*, pp. 199-223, 1997.

[18] Myron Flickner, et al., "Query by Image and Video Content: The QBIC System," *IEEE Computer*, Vol.28, No.9, pp. 23-32, 1995.

**류 은 숙**

1990년 2월 충남대학교 공과대학 컴퓨터공학과(학사)
1992년 2월 충남대학교 공과대학 컴퓨터공학과(석사)
1998년 8월 충남대학교 공과대학 컴퓨터공학과(박사)
관심분야 : 멀티미디어 데이터베이스 시스템, 객체지향 데이터베이스 시스템, 전자 도서관

**이 규 철**

1984년 2월 서울대학교 공과대학 컴퓨터공학과(학사)
1986년 2월 서울대학교 공과대학 컴퓨터공학과(박사)
1989년 3월~현재 충남대학교 공과대학 컴퓨터공학과 부교수
관심분야 : 데이터베이스 시스템, 멀티미디어 시스템, 소프트웨어 공학, 분산 처리