무선 멀티미디어 서비스 환경에서 MPEG-4 스트림을 위한 객체 관리기의 설계†

(Design of Object Manager for MPEG-4 Stream in the Wireless Multimedia Service Environment)†

최숙영*
(Sook-Young Choi)

요 약 현재 멀티미디어 서비스 분야에서 핵심적인 기술로 부상되고 있는 MPEG-4는 고압축률을 제공하며, 그 자체 구조의 모든 요소을 기술하는데 객체지향적인 기법을 사용하고 있다. 본 연구에서는 다양한 통신망과 단말 환경에 잘 대응하고 강력한 상호작용을 지원함으로서 효과적인 멀티미디어 서비스를 제공할 수 있도록, MPEG-4 시스템에 객체 관리기의 기능을 추가하도록 한다. 이 객체 관리기를 통해, 장면트리를 구성하는 객체들에 우선 순위를 부여하고 관리하며, 사용자 상호작용에 따른 장면트리의 변경 및 각 객체의 정보 및 버젼들을 관리하도록 한다. 또한 객체간의 시간 관계 모델을 정의함으로서 동기화를 효과적으로 지원할 수 있도록 한다.

핵심주제어 : MPEG-4 시스템, 객체 관리기

Abstract MPEG-4 provides high compression rate and uses object-oriented method to describe components of its structure, which has currently risen as the core technique in multimedia service fields. Our research objective is to provide object manager to MPEG-4 system in order that effective multimedia service could be available by supporting powerful interaction and adapting to various networks and terminals. Through the object manger, priorities are given to objects of a scene tree and the objects having higher priorities are first rendered according to the terminal capability. It also manages synchronization and update of the scene tree and object informations caused by user interactions.

Key Words : MPEG-4, Object Manager

1. 서 론

MPEG-4는 현재 멀티미디어 서비스 분야에서 핵심적인 기술로 부상되고 있다. MPEG-4는 고압축률을 제공하며, 그 자체 구조의 모든 요소을 기술하는데 객체지향적인 기법을 사용하고 있다[1]. 이러한 MPEG-4의 객체기반의 부호화 방식은, 복수개의 객체를 합성하여 한 장면에 표현하고 있으며, 각 장면을 트리 형태로 나타내고 있다. 사용자가 장면의 일부 또는 전체를 지정된 시간 및 이벤트에 따라서 갱신할 수 있는 방법을 제공함으로서 강력한 사용자 상호작용을 지원할 수 있도록 한다. 또한 MPEG-4는 저비트율의 전

† 본 연구는 2001학년도 정보통신부의 대학기초 연구 지원에 의해 수행되었음
* 우석대학교 컴퓨터교육과

송에서도 고품질의 동영상을 제공할 수 있고, 오류 내성이 강하기 때문에 전송 오류률이 높은 이동 통신에서도 적합하다고 할 수 있다[2].

인터넷 및 협대역 통신망에서 효과적인 멀티미디어 서비스를 제공하기 위해서는 스케일러블(scalable) 전송이 요구되며, MPEG-4에는 이를 위해 시.공간 스케일러빌러티(scalability)를 지원하고 있다[3]. 이러한 시.공간 스케일러빌러티는 MPEG-2에서 표준화된 것과 거의 같은 기능을 지원하고 있으며, MPEG-4의 객체 기반의 부호화의 특징을 잘 반영한 스케일러빌러티는 아니다.

핸드폰이나 PDA와 같은 무선 통신 환경에서 좀더 효과적인 MPEG-4기반의 멀티미디어 서비스를 제공하기 위해서, MPEG-4의 객체기반 부호화의 특징을 이용한 다른 차원의 스케일러빌러티가 고려될 수 있다. 즉, MPEG-4의 디코드 부분에서 장면트리를 구성하는 객체들에게 우선 순위를 부여하여, 단말기의 성능이 좋지 않을 때는 우선 순위가 낮은 객체를 장면트리에서 제외시킨 다음 랜더링하여 프리젠테이션하고, 성능이 우수한 단말기인 경우에는 장면트리를 구성하는 모든 객체들을 프리젠테이션하도록 함으로서, 단말기에 따라서 이에 적합한 스케일러블 서비스를 제공할 수 있다.

또한 사용자 상호작용으로서 사용자가 장면을 구성하는 객체를 삭제하거나, 새로운 객체를 추가하고, 한 객체의 내용을 변경할 경우, 이에 따르는 장면트리를 관리하고, 각 객체들을 효과적으로 관리하는 일이 요구된다. 특히, 객체의 내용이 변경되었을 경우, 그 객체에 대한 버젼(version) 관리도 매우 중요하다고 할 수 있다. 이러한 객체 관리가 적절히 지원되지 않을 경우에는 프리젠테이션에 이상이 생길 수 있고, 객체간의 충돌이 발생되어 효과적인 상호작용을 지원 할 수 없다. 한편, MPEG-4에서는 객체간의 동기화를 위해 타임스탬프를 사용하고 있다. 하지만, 사용자 상호작용이 발생할 때는 객체내의 시간 관계가 정의되어 있지 않아 동기화의 문제가 발생될 수 있다[5].

본 연구에서는 MPEG-4 플레이어에 객체 관리기의 기능을 추가하고, 객체간의 시간 관계 모델을 정의함으로서, 다양한 통신망과 단말 환경에 따라 적합한 멀티미디어 서비스를 하기 위한 스케일러빌리티와 강력한 사용자 상호작용을 효과적으로 지원할 수 있도록 한다.

2. 관련 연구

2.1 MPEG-4의 특징

기존의 부호화 표준 특히, MPEG-1이나 MPEG-2와 비교하여 MPEG-4의 특징은 크게 객체 기반, 합성 AV 처리, 객체별 대화성 제공 등으로 요약될 수 있다[4,5]. 첫째, 객체 기반의 부호화 기술의 관점에서 살펴보면, MPEG-1, MPEG-2, H.261, H.263 등의 이제까지의 부호화 표준이 AV(Audio/Visual) 정보를 장면(프레임)을 기준으로 부호화하는데 반하여, MPEG-4는 AV정보를 오디오/비디오 오브젝트로 분리하고 이들 객체를 부호화한다. 따라서, 장면의 재생시에는 하나 하나 독립적으로 부호화된 객체들을 역시 별도로 복호화한 후에 이들을 합성하여 하나의 장면을 구성하고 화면에 표시한다. 이와같이 MPEG-4에서는 AV정보들을 객체 단위로 부호화하기 때문에 기존의 부호화 표준에서는 불가능하였던 객체 단위의 조작, 가공 및 편집 등이 가능하며, 방송, 인터넷, DVD 등의 팩키지 미디어 분야에서 멀티미디어 컨텐츠의 제작/편집 등이 적합한 기능을 제공한다. 둘째는 컴퓨터에 의하여 생성된 합성 영상과 합성 음까지도 부호화의 대상으로 하고 있는 것이다. 즉, MPEG-1과 MPEG-2에서는 카메라 및 마이크로부터 취득한 자연 영상과 자연 음만을 대상으로 하여 부호화하였지만, MPEG-4는 컴퓨터그래픽에 의하여 생성된 VRML, 2차원 및 3차원의 그래픽 모델, 얼굴 애니메이션, 텍스트 등의 합성 영상과 MIDI 등의 합성 음도 부호화 한다. 셋째는, AV 정보의 내용에 기반을 둔 대화적 기능의 제공이다. MPEG-4는 AV정보를 모아서 하나의 장면을 구성하기 때문에 그 장면을 구성하거나 표현할 때 사용자가 대화적인 방법에 의해서 원하는 형태로 장면을 구성하거나, 표현할 수 있도록 하는 기능을 제공한다.

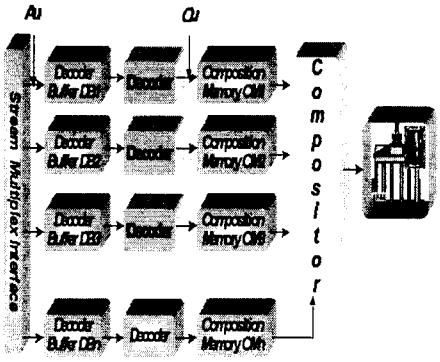
2.2 MPEG-4 시스템

2.2.1 MPEG-4 시스템 디코더 모델

MPEG-4는 객체에 기반을 두고 있기 때문에 수신 단말에 장면 구성기(compositor)가 포함되

어 있으며, 이러한 장면 구성을 위하여 디코딩 버퍼 외에 기존의 다른 수신 단말에 없는 컴퍼지션 버퍼를 추가로 갖는다[5]. 이들 버퍼의 동기는 각각 DTS(Decoder Time Stamp) 와 CTS(Compositor Time Stamp)를 이용하여 이루어진다. <그림 1>은 MPEG-4의 시스템 디코더 모델(SDM : System Decoder Model)을 나타낸 것이며, 구성 요소를 설명한다.

MPEG-4 시스템에서는 다중화에 기존의 방식을 사용하기 위하여 DMIF(Delivery Multimedia Integration Framework)가 다중화 층과의 인터페이스를 제공한다. 기초 스트림(ES : Elementary Stream)은 개별 부호화 비트 스트림, MPEG-4 비주얼 등의 인코더 출력 및 디코더 입력이 이에 해당된다. 접근 단위(AU : Access Unit)는 ES를 분할하여 얻는다. 복호/합성을 위한 시간관리 및 동기를 위한 처리 단위가 된다. MPEG-1,-2 비디오는 픽셀에, MPEG-4 비주얼에서는 VOP 부호화 데이터에 해당한다. 복호화 버퍼(DB : Decoding Buffer)는 AU를 보존하는 버퍼로 크기는 송출 측에서 지정하는 것도 가능하다. 디코더(Decoder)는 AU를 입력으로 하여 컴포지션 유닛(CU: Composition Unit)를 출력한다. MPEG-4 비주얼 디코더 등이 이에 해당한다. 컴포지션 메모리(CM : Composition Memory)는 합성을 위하여 CU를 보존하는 메모리로 임의 접근이 허용되고 있으며 크기는 규정하지 않는다. 컴포지터(Compositor)는 장면기술을 기초로 CU를 합성한다.

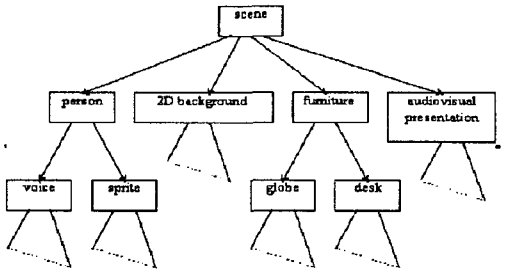


<그림 1> MPEG-4 시스템 디코더 모델

MPEG-4 시스템의 주요 기능으로는 멀티미디어 컨텐츠의 표현에 관련된 기능으로 장면 기술, 장면과 스트림과의 연결, 장면 애니메이션을 들 수 있다. 또한 비트 스트림 관리에 관련된 기능으로 타임과 버퍼 관리, 동기화, 기본 스트림의 다중화/역다중화을 들 수 있다.

2.2.2 장면 기술(Scene Description)

MPEG-1과 MPEG-2에는 장면이라는 개념이 없었다. MPEG-2까지는 구형형상의 비디오 부호화만을 취급해 왔기 때문이다. 한편, MPEG-4에서는 임의형상의 객체를 부호화할 수 있다. 또 압축 부호화한 비디오 이외에도 파라미터를 지정하여 생성되는 객체도 취급할 수 있게 되었다. 시청자에게 제시되는 화면은 MPEG-2까지는 하나의 비디오로 구성되는 장면에서 복수의 객체를 합성한 장면으로 변한다. 이 때문에 MPEG-4 시스템에서는 객체의 표시 방법과 특성을 지정하기 위한 장면기술 언어로 BIFS(Binary Format for Scene)를 규격화하고 있다[5].



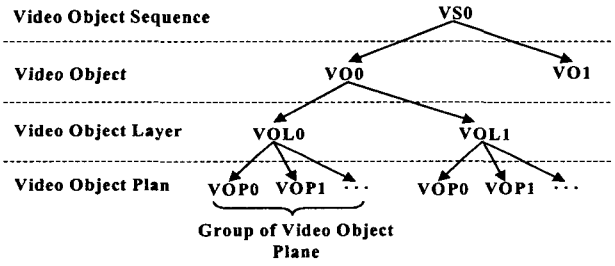
<그림 2> 장면의 논리적 구조의 예

장면 구성을 위한 정보는 BIFS에 기술되며, A/VO의 압축 스트림을 전송하기 전에, 객체에 관한 정보를 담고 있는 객체 서술자 (OD : Object Descriptor)들과 우선적으로 전송되어야 한다. <그림 2>는 장면의 논리적 구조의 예를 나타낸다. BIFS는 각 객체들의 그룹화, 시공간상의 위치 지정, 특성 값 선택, 모양, 텍스쳐, 피치 등의 변환 파라미터 등을 포함한다.

2.2.3 스케일러빌리티(Scalability)

스케일러빌리티는 인터넷 및 대역폭이 적은 무

선망에서의 비디오 전달, 비디오 데이터베이스 브라우징에 중요한 기능이다[3]. 특히 현재의 인터넷에서는 전송대역에 대한 네트워크의 서비스 품질(QoS : Quality of Service)이 보증되고 있지 않으므로, 동영상을 높은 부호화 속도로 안정하게 전송하기가 쉽지 않다. 아울러 동영상 신호를 소프트웨어로 처리하는 것이 일반화되었지만, 처리 능력이 낮은 단말에서는 수신한 부호화 데이터를 완전히 복호 할 수 없는 경우도 문제이다. 따라서, 저해상도와 고해상도의 동영상을 준비해 놓고, 네트워크와 단말의 상태가 양호할 때는 고해상도의 동영상을 복호하고, 상태가 악화되었을 경우는 화질을 저하시키는 것이 아니라, 저해상도의 품질을 보충하는 것을 생각할 수 있다. MPEG-4 비디오는 이러한 상황을 고려하여 시공간 스케일러빌리티의 기능을 제공하고 있다. <그림 3>은 스케일러빌리티의 기능을 설명하기 위하여 MPEG-4 비디오의 신텍스 계층(syntax layer)을 설명하고 있다.



<그림 3> MPEG-4 Video의 신텍스 계층

동영상은 복수의 비주얼 객체(VO : Video Object)로 구성되며, VO는 복수의 비디오 객체 레이어 (VOL : Video Object Layer) 데이터로 구성되며, VOL은 복수의 비디오 객체 평면 (VOP : Video Object Plane) 데이터로 구성되어 있다[2]. 프레임에 해당하는 VOP와 VO 사이에 VOL 구조가 정의되어 있는 것은 시공간 스케일러빌리티를 표현하기 위함이다. 여기서 VOL0으로 전송된 정보로부터 저해상도 영상을 얻고, VOL0+VOL1로 보내진 정보로부터 고해상도 영상을 얻는다.

3. 동기화 모델

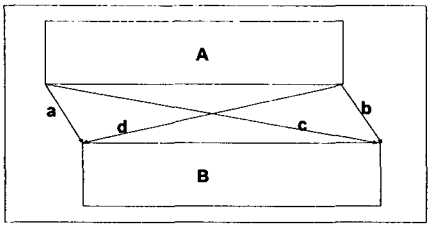
멀티미디어 프레젠테이션에서 미디어의 함축된 의미를 효과적으로 전달하기 위해서는 미디어 데이터간에 존재하는 시간적인 정보를 효율적으로 표현하고 처리할 수 있는 동기화 (synchronization) 기법이 필요하다[10]. 그런데, MPEG-4의 시간 관계 모델은 타임스탬프(time stamp)와 참조 클럭(reference clock)에 기초한 것으로서 미디어내(intra-stream)의 동기화는 정확히 제공할 수 있지만, 미디어간(inter-stream)의 동기화는 효과적으로 지원할 수 없다는 문제점이 존재한다[5]. 특히, 사용자 상호작용으로서 미디어가 추가되거나 삭제될 경우, 미디간의 동기화 문제가 발생될 수 있다. 그러므로, 본 연구에서는 MPEG-4 미디어간의 시간관계를 표현하고 동기화를 효과적으로 처리할 수 있는 시간 관계 모델을 제안한다.

3.1 인과성 시간 관계

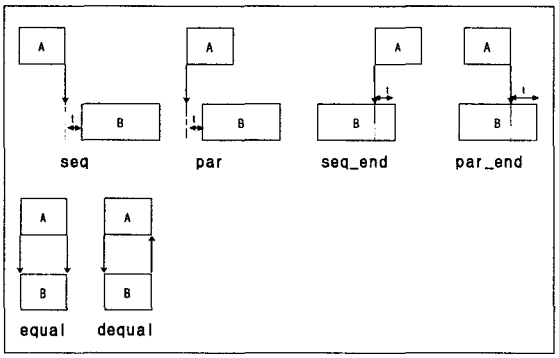
본 연구에서는 인과성 관계를 바탕으로 시간 관계를 표현한다. 인과성 관계는 이유, 목적, 양보, 의뢰로써[11], 프리젠테이션 상에 실행되는 미디어들은 서로간에 어떤 이유나 목적에 의해 서로 관련성을 맺게 된다. 하지만 다른 미디어들과 아무런 이유나 목적 관계가 존재하지 않을 경우에는 사용자에 의해서 미디어간에 양보 혹은 의뢰에 의해 프리젠테이션에 존재할 수 있다. 이는 실제 저작 시나리오에도 나타날 수 있는 현상으로 완전히 독립적인 수행을 하는 미디어에 대해서 인과성 관계를 줄 수 있다는 의미이다.

인과성을 표현하는 시간 관계는 하나의 객체(A)에 존재하는 두 개의 동기점인 시작점($\underline{A}$), 끝점($\bar{A}$)을 사용해서 그림 4와 같이 크게 4가지로 구성할 수 있다. 하나의 객체에 대한 두 개의 동기점을 나타낸다. 시작점은 한 객체의 시작을 나타내고, 끝점은 한 객체의 멈춤을 나타낸다. 이를 수식으로 표현하면 구간 A의 시작점 $\underline{A}$와 끝점 $\bar{A}$ 사이의 시간은, $\underline{A} \leq \bar{A}$ 으로 나타내고, $\bar{A} - \underline{A}$값이 구간 A의 미디어 실행 시간 값이 된다.

$$\text{실행 시간}(A) = \bar{A} - \underline{A} \qquad \text{식 1.1}$$



<그림 4> 객체간의 인과성 관계



<그림 5> 인과성 시간 관계

<그림 4>는 두 개의 객체 A, B 간에 존재하는 인과성 관계를 표현한 것이다. A, B 각 동기점에 의해서 한 객체는 cause 객체가 되고, 다른 객체는 effect 객체가 된다. 이런 관계는 화살표 방향에 따라 그 역할이 결정된다. 즉 cause 이벤트를 발생하는 A 객체는 능동 객체의 역할을 수행하고, effect 이벤트가 발생하는 다른 한 객체는 수동 객체의 역할을 수행하는 것이다. 4가지 관계를 설명하면, a는 A가 시작하면서 B를 시작시키는 관계, b는 A가 끝나면서 B를 끝내는 관계, c는 A가 시작하면서 B를 끝내는 관계, 마지막으로 d는 A가 끝나면서 B를 시작시키는 관계를 나타낸다.

<그림 4>를 바탕으로 본 연구에서 정의한 인과성 기반 시간 관계는 <그림 5>에 나타냈다. 기본적인 4가지 인과성 관계와 지연시간을 포함하여 seq, par, seq_end, par_end를 정의했고, 추가적으로 동일성(equality) 관계를 equal, dequal로 정의했다. 그 이유는 미디어 객체의 수행시간이 외부 이벤트에 의하여 변경될 때, 서로 인과성 시간 관계에 있는 객체끼리 끝점을 지정하기 위함이다. 본 연구에서 지연시간은 인과성 있는 객체간에 존재하며 능동 객체와 수동 객체간의 원만한 시나리오 수행을 위해서 지연되는 시간을 말한다. <그림 4>의 6가지 시간 관계에 대한 명세를 <표 1>에 정의하였다.

<표 1> 인과성 시간 관계 문법 및 의미

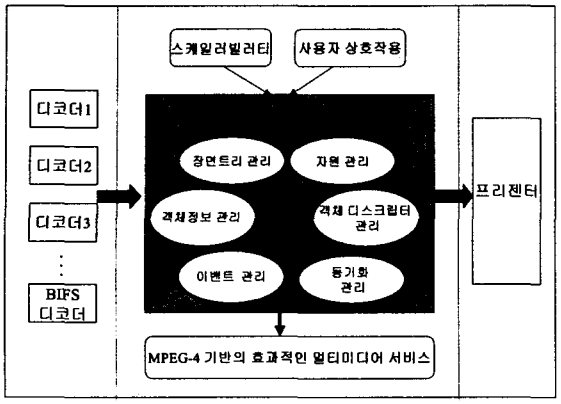
문법	의미
A seq(t) B	구간 A의 끝점은 t 지연시간 후에 구간 B를 시작하게 한다.
A par(t) B	구간 A의 시작점은 t 지연시간 후에 구간 B를 시작하게 한다.
A seq_end(t) B	구간 A의 시작점은 t 지연시간 후에 구간 B를 끝나게 한다.
A par_end(t) B	구간 A의 끝점은 t 지연시간 후에 구간 B를 끝나게 한다.
A equal B	구간 A의 시작점은 구간 B를 시작하게 하고, 구간 A의 끝점은 구간 B를 끝나게 한다.
A dequal B	구간 A의 시작점은 구간 B를 시작하게 하고, 구간 B의 끝점은 구간 A를 끝나게 한다.

4. 객체 관리기의 설계

4.1 객체 관리기의 구성

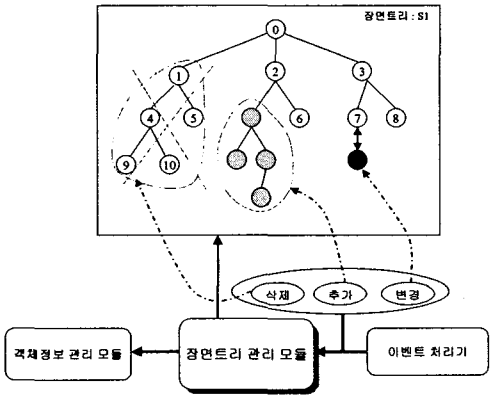
객체 관리기는 핸드폰이나 PDA 같은 무선통신 환경에서 단말기의 성능에 따라 적절하게 멀티미디어 서비스를 지원 할 수 있도록 객체의 우선순위를 이용하여 스케일러빌리티를 지원하고, 장면을 제어하는 노드 개체들간의 융통성 있는 시간 관계와 시간관계에 기초한 동기화 트리를 제공함으로서 강력한 사용자 상호작용을 지원하고 있다.

본 연구에서 제안하는 객체 관리기의 구성은 <그림 6>과 같다. 객체 관리기는 스케일러빌리티와 사용자 상호작용을 지원하기 위해 장면트리 관리 모듈, 객체 우선순위와 버전을 관리하는 객체정보 관리 모듈, 객체 스크립터 관리 모듈, 이벤트 관리 모듈, 동기화 관리 모듈들로 구성되며, 이를 통해 MPEG-4 기반의 효과적인 멀티미디어 서비스를 제공할 수 있다.



<그림 6> 객체 관리기

4.1.1 장면트리 관리 모듈



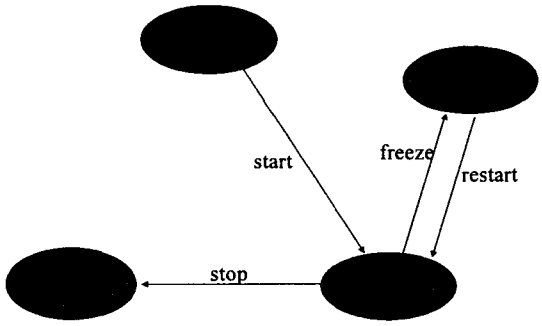
<그림 7> 사용자 상호작용에 대한 장면트리의 변화

MPEG-4 시스템 부분에서는 객체의 표시 방법과 특성을 지정하기 위한 장면 기술 언어로서 BIFS(Binary Format for Scene)를 규격화하고 있다. 이러한 BIFS는 VRML을 바탕으로 하고 있다. 장면은 트리 상태로 배치되어 다양한 노드들의 집합으로 표현되고 있다.

장면트리 모듈은 장면 기술 스트림과 각 노드들의 특성에 따라 분류해 놓은 NDT(Node Data Type), 각 노드들이 포함하고 있는 필드들에 대한 정보 NCT(Node Codeing Table) 들을 참조하여 장면트리를 구성한다. 또한 장면트리에 사용자 상호작용으로서 새로운 객체를 추가하거나 삭제, 변경하는 경우 이를 관리하게 된다. <그림 7>은 노드의 추가, 삭제, 변경으로 인한 장면 트리의 변경을 보여준다.

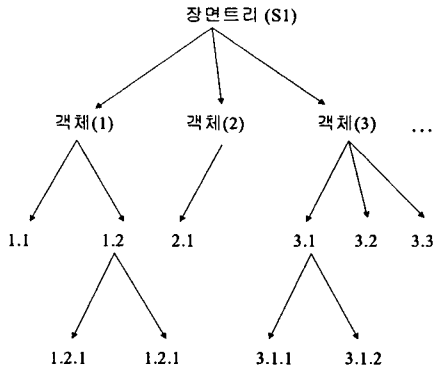
4.1.2 객체정보 관리 모듈

객체정보 관리 모듈에서는 장면트리를 구성하고 있는 객체들의 우선 순위와 버전을 관리한다. 객체들의 우선순위는 장면트리가 생성이 될 때 기초 스트림의 디스크립터에 초기에 부여된 우선순위를 가지거나, 사용자가 우선순위를 부여할 수 있도록 한다. 이 모듈에서는 사용자 상호작용으로 객체의 추가, 삭제, 변경이 발생하면 객체정보 관리 모듈에서는 객체들의 우선순위를 재조정하게 된다. 객체들의 우선순위를 유지함으로서, 이 우선순위에 따라 장면을 재생시키게 된다. 객체들의 우선순위를 이용하여 단말기의 성능에 따라 적절하게 객체들을 재생시킬 수 있다. 즉, 단말기의 성능이 좋으면 장면트리를 구성하는 객체들을 모두 재생시키고, 단말기의 성능이 좋지 않을 때는 우선순위가 높은 객체들만 재생시킨다. 따라서, 객체수준의 스케일러빌러티를 지원할 수 있다.



<그림 8> 미디어 객체의 상태 변화

또한 객체정보 관리 모듈에서는 객체들의 버전을 관리하게 되는데, 초기에 부여된 책체들의 ID를 이용하여 객체에 변경이 발생하였을 경우 버전의 이름을 부여한다. 이 버전들을 이용하여 변경된 객체의 정보를 효과적으로 파악할 수 있다. 또한 장면트리에도 ID를 부여하여 장면들의 변경 정보를 쉽게 파악하고 이를 적절히 이용할 수 있도록 한다. <그림 8>은 한 장면 트리를 구성하는 객체들의 버전을 관리 하는 트리이다.



<그림 9> 버전 관리 트리

한편, 객체 정보 관리 모듈은 실행시 미디어 객체들의 상태를 관리하며, 사용자 상호 작용이 발생될 경우 그 객체의 상태 정보를 이용하여 처리한다. 실행시 변화되는 객체들의 상태 전이도는 <그림 9>와 같다.

<그림 9>와 같이 객체들의 상태는 크게 idle, ready, complete, run으로 구성되며, 실행시간의 어느 한 시점에서 이 4 가지 상태 중 한 상태를 유지하게 된다. Idle은 객체의 초기 상태, ready는 지연시간 및 일시 정지를 위한 대기 상태, run은 디스플레이되는 실행 상태, complete는 객체의 수행 완료 상태를 나타낸다. 위의 상태 전이도에서 간선들은 발생되는 이벤트들을 나타내며, 각 이벤트가 발생하는 경우 객체들의 상태 변화를 볼 수 있다.

4.1.3 객체 디스크립터 관리 모듈

장면 기술에서 비디오 및 오디오 스트림을 참조하는 노드는 객체 디스크립터 식별자라고 부르는 OD_ID를 지정하며, 객체 스크립터는 노드에 지정된 스트림의 정보로서 ES 디스크립터, OCI(Object Content Information) 디스크립터 등을 포함한다. ES 디스크립터는 비디오 및 오디오 스트림이 스케일러블 부호화되어 복수의 스트림으로 나뉘어지는 경우 각각의 스트림에 대해서 필요하게 된다. 각 스트림의 ES 디스크립터는 ES_ID에 의하여 식별된다. OCI 디스크립터에는 스트림의 내용을 분류하고, 키워드 및 등급 정보의 부여, 저작권 명칭등을 기술한다.

객체 디스크립터 관리 모듈은 새로운 스트림이 추가될 때 이 객체를 식별하기 위한 ID를 부여하고, 삭제되었을 경우에는 해당 객체 디스크립터를 삭제한다. 또한 객체 디스크립터내의 ES 디스크립터가 변경될 경우에도 새로운 ES 디스크립터를 유지하도록 한다. 특히, 한 객체 디스크립터내에 row-bit rate 스트림을 위한 ES 디스크립터와 high-bit rate 스트림을 위한 ES 디스크립터로 구성되어 있는 경우, 현재 시스템에서 지원가능한 프레임률에 따라 적합한 ES를 선택하여 재생하도록 한다.

4.1.4 이벤트 관리 모듈

이벤트 관리기 모듈은 사용자 상호작용에 의한 이벤트를 감시하고, 이벤트가 발생되면, 이벤트가 발생된 노드에 대한 id를 저장하고 해당 작업을 처리한다. 또한 사용자 상호작용에 의해 프리젠테이션 순서가 변환되어 장면을 구성하는 객체간의 시간관계를 변경해야 될 경우, 이에 대한 처리를 위해 동기화 관리 모듈에 통보한다.

4.1.5 동기화 관리 모듈

동기화 관리 모듈은 먼저, 각 객체들의 디스크립터에 있는 ES 디스크립터를 참조하여 각 스트림들의 객체별 시간 기준(OTB:Object Time Base)을 구한 후, 이것들을 기초로 하여 객체 스트림들간의 시간 관계를 정의한다. 이때 시간 관계는 3 장에서 기술된 인과성 시간 관계를 바탕으로 표현된다. 또한 사용자 상호작용이 발생될 경우, 이벤트 관리 모듈로부터 신호를 받고 미디어간의 시간관계를 조정한다.

4.1.6 자원 관리 모듈

시스템의 상태를 체크하는 모듈로서 단말기의 성능을 점검 후, 이에 대한 정보를 장면트리 관리 모듈과 객체정보 관리 모듈에게 제공한다. 이 경우, 장면 트리 관리 모듈과 객체 정보 관리 모듈은 이 정보를 토대로 객체의 우선 순위와 장면 트리의 다른 버전을 고려하여 적절하게 수행될 수 있도록 제어한다.

4.2 객체 관리기의 처리 과정

이벤트 관리 모듈은 실행시간에 사용자로부터

오는 상호작용을 감시하다가 한 이벤트가 발생되는 경우, 각 모듈에게 신호를 보낸다. 각 모듈들은 사용자 상호 작용에 대한 신호를 받고 이에 대한 처리를 시작한다. 다음은 각 이벤트에 대한 처리 루틴을 나타낸다.

Procedure Event_process()

/* i : *a media object*, k : *one of related objects with object i*, E_t : *current time at the point of event occurred*, S_t(i) : *presentation start time of i*, F_t(i) : *presentation finish time of i*, D(i) : *duration of i*, D_r(i) : *remaining presentation time for i*, Sk_t : *time after skip*, Pt_g : *presentation time gap as scaling the speed*, V : *speed of presentation*, △V : *scaling factor of presentation speed*, D_r'(i) : *remaining presentation time for i as the result of scaling event*, FZ : *freeze*, RS : *resume*, F_SK : *forward skip*, B_SK : *backward skip*, SS : *scaling* */

```
begin
  case event-type : FZ
    for all objects that the current object state is run
    send STOP signal to the media players for media
    change the object state from run to ready
     compute the duration with remaining time
   D_r(i) = D(i) - |E_t - S_t(i)|
  endcase

  case event-type : RS
   for all objects that the current object state is ready,
   send START signal and D_r(i) to the media players
   change the object state from ready to run
  endcase

  case event-type : F_SK
   for all objects that the current object state is run,
   send STOP signal to the media players
   change the object state from run to ready
      compute the skip time
   check the objects to be processed after the skip
      time
   compute the duration with remaining time
   if (S_t(i) < Sk_t) and (F_t(i) > Sk_t) then
      if (S_t(i) < E_t) then
         D_r(i) = D(i) - (|S_t(i) + E_t| + skip time )
      endif
      if (S_t(i) > E_t) then
         D_r(i) = D(i) - (skip time - |S_t(i) - E_t|)
      endif
   endif
      if (D_r(i) < skip time) then
        change the object state from ready to Complete
        for the objects to be processed after the skip
           time,
        send START signal and D_r(i) to the media
           players
        change the object state from ready or idle to
           run
  endcase

  case event-type : B_SK
     for all objects that the current object state is run,
        send STOP signal to the media players
     change the object state from run to ready
     compute the skip time
     check the objects to be processed after the skip
      time
     compute the duration with remaining time
      if (S_t(i) < Sk_t) and (F_t(i) > Sk_t) then
        if (F_t(i) > E_t) then
          D_r(i) = skip time + (D(i) - (|E_t - S_t(i)|))
        endif
        if (F_t(i) < E_t) then
          D_r(i) = D(i) - (|(E_t - skip time) -|S_t(i)|)
        endif
      endif
     for the objects to be processed after the skip
        time,
       send START signal and D_r(i) to the media
        players
       change the object state from ready or idle to
        run
  endcase

  case event-type : SS
     for all objects that the current object state is run,
       reset duration
       D_r'(i) = D_r(i) * V * △V
       Pt_g = D_r(i) - D_r'(i)
       D_r'(k) = D_r(k) + Pt_g
   endcase
end
```

<그림 10> 사용자 상호작용에 대한 수행 알고리즘

4.2.1 일시정지 및 재수행(freeze and resume)

사용자가 프리젠테이션의 수행을 일시정지하라

는 명령을 입력하면, 프리젠테이션 관리자는 현재 프리젠테이션 중인 모든 미디어를 중단시키기 위해 현재 객체 상태가 run인 모든 미디어 객체에 대한 미디어 플레이어에게 STOP 신호를 보내고, 객체 상태를 run에서 ready로 변경한다. 잔여시간으로 미디어의 수행시간을 고정시킨다.

사용자가 재수행 명령을 입력하면, 현재 일시정지 상태에 있는 모든 객체의 수행을 재개하기 위해 현재 객체 상태가 ready인 모든 미디어 객체에 대한 미디어 플레이어에게 START 신호를 보내고, 객체 상태를 ready에서 run으로 변경한다.

4.2.2 전진방향 스킵(forward skip)

사용자가 현재 프리젠테이션 중인 미디어의 수행을 전진방향 스킵하라는 명령을 입력하면, 현재 수행중인 모든 미디어 객체들에 대한 미디어 플레이어에게 STOP 신호를 보내 프리젠테이션을 정지시키고 객체 상태를 run에서 ready로 변경한다. 스킵시간을 계산한 후, 스킵시간이 지난 다음에 프리젠테이션 되어야 할 미디어 객체들을 체크하여 각 미디어들의 수행시간을 남은 시간으로 갱신한다. 또한 그 미디어들을 수행시키기 위해 그 미디어 플레이어들에게 START 신호를 보내고, 객체들의 상태 정보를 ready에서 run으로 혹은 idle에서 run으로 변경한다.

4.2.3 후진방향 스킵(backward skip)

현재 수행중인 모든 미디어 객체들에 대한 미디어 플레이어에게 STOP 신호를 보내 프리젠테이션을 정지시키고 객체 상태를 run에서 ready로 변경한다. 스킵시간을 계산한 후, 스킵시간 이전에 프리젠테이션 되어야 할 미디어 객체들을 체크하여 각 미디어들의 수행시간을 갱신한다. 또한 그 미디어들을 수행시키기 위해 그 미디어 플레이어들에게 START 신호를 보내고, 객체들의 상태 정보를 ready에서 run으로 혹은 idle에서 run으로 변경한다.

4.2.4 프리젠테이션 속도 조절(scaling of presentation speed)

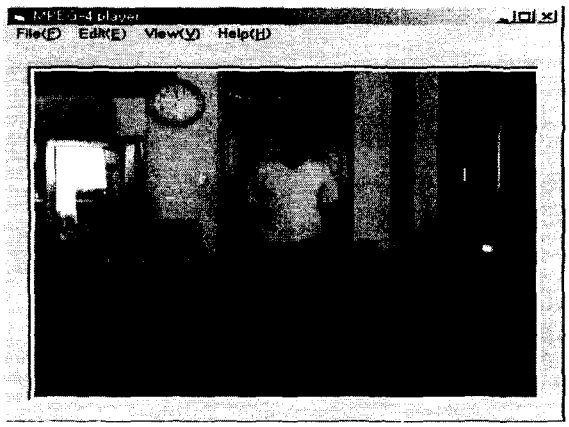
사용자가 프리젠테이션 수행 중인 미디어의 프리젠테이션 속도를 늦추거나 빠르게 하는 프리젠테이션 속도 조절 명령을 입력하면, 지정된 시간 범위에서 수행되어야 할 모든 미디어들을 체크하여 그 미디어의 수행 속도를 변화시킨다. 또한, 지정된 시간 뒤에 수행될 예정인 미디어의 속도는 변경되지 않지만 동기화를 위해 프리젠테이션 속도의 조절비에 따라 프리젠테이션 시간을 연장 또는 단축시켜야 하는 등의 프리젠테이션 변화가 반영된다.

5. 실험 분석

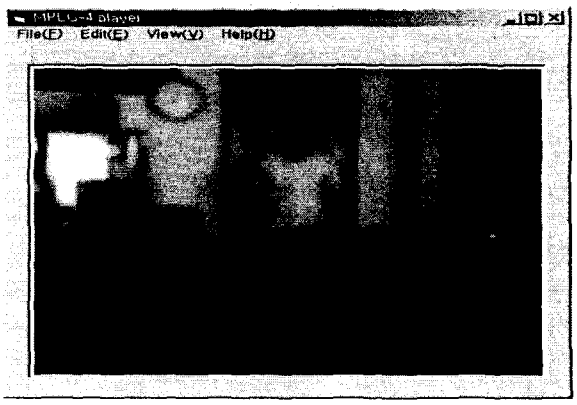
본 연구의 실험 분석으로 스케일러빌리티와 사용자 상호작용의 두가지 관점에서 수행되었다.

5.1 스케일러빌리티의 관점

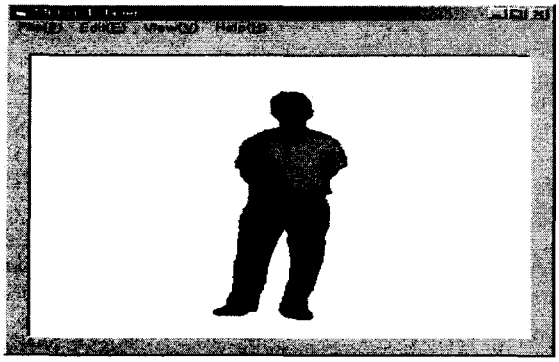
스케일러빌리티의 관점에서 실험분석은 단말기의 성능에 맞게 객체의 우선 순위와 기초 스트림의 비트율을 고려하여 랜더링되는 것을 보여주는 것이다.



<그림 11> MPEG-4 파일1의 랜더링



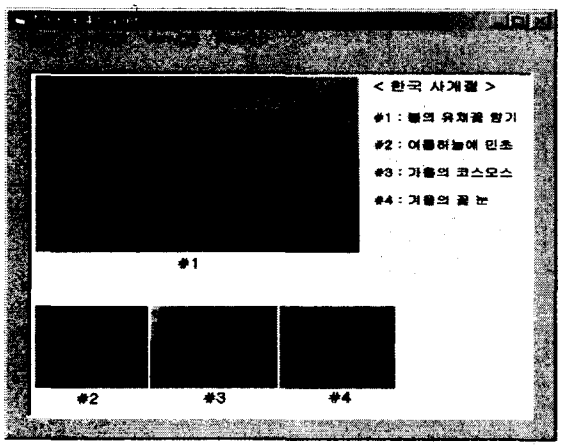
<그림 12> 저비트율의 스트림을 랜더링한 화면



<그림 13> 객체의 우선순위를 고려하여 랜더링한 화면

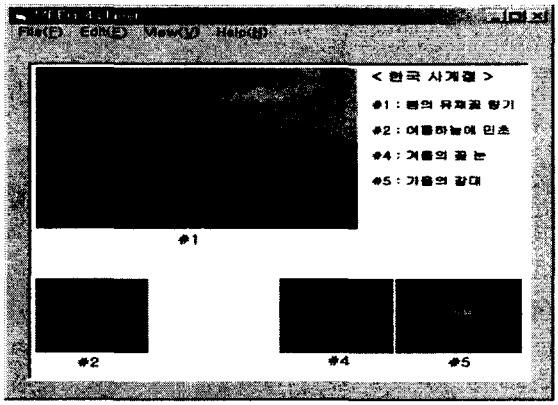
<그림 11>은 원래의 MPEG-4 파일을 랜더링 한 것이고, <그림 12>는 저비트율의 기초스트림을 랜더링한 것을 보여준다. 또한 <그림 13>은 객체의 우선 순위를 고려하여 장면을 구성하는 노드 객체들 중 우선 순위가 낮은 객체는 제외시키고 랜더링 한 것으로서 배경을 제외시키고 전경에 있는 사람만을 랜더링 한 것을 보여준다.

5.2 사용자 상호작용성의 관점

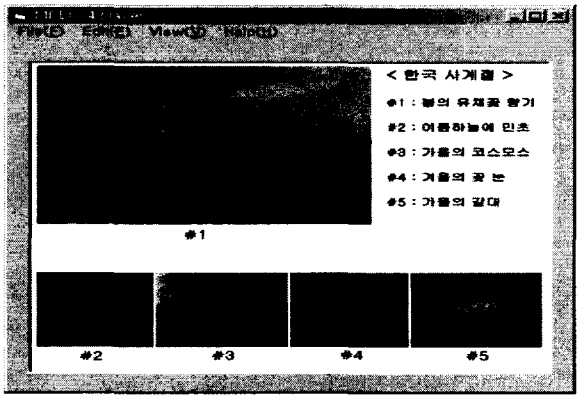


<그림 14> MPEG-4 파일2의 랜더링

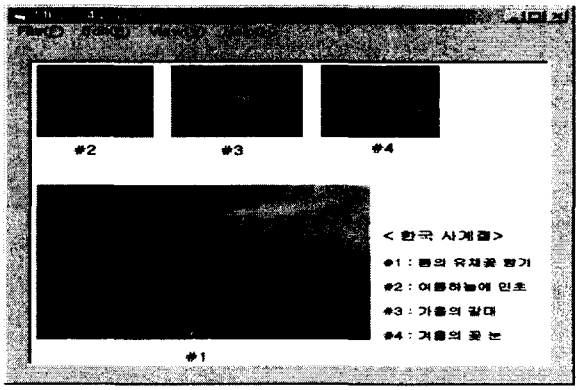
사용자 상호작용에 대한 실험 분석으로 장면을 구성하는 노드들 중에서 하나의 노드를 제거하거나, 추가, 변경하는 것을 보여주는 것이다. <그림 14>는 원래 MPEG-4 파일을 랜더링한 것을 보여주고, <그림 15>는 사용자 상호작용에 의해 한 객체를 삭제한 것을 보여준다. <그림 16>은 원래의 파일에 새로운 객체를 추가한 것을 보여준다. 또한, <그림 17>는 장면을 구성하는 객체들의 속성(위치값)을 변경한 것을 보여준다.



<그림 15> 한 객체를 삭제한 경우



<그림 16> 한 객체를 추가한 경우



<그림 17> 객체들의 위치를 변경한 경우

6. 결 론

본 연구에서는 다양한 통신망과 단말 환경에 잘 대응하고 강력한 상호작용을 지원하는 효과적인 멀티미디어 서비스를 제공할 수 있도록 MPEG-4 시스템에 객체 관리기의 기능을 추가하고 있다. 이를 위해 장면을 구성하는 BIFS를 분석하여 장면트리를 구성하고 관리하는 장면 관리 모듈과 장면트리를 구성하는 각 노드 객체들을 관리하는 객체정보 관리 모듈들을 설계하였다. 또한 스케일러빌리티를 지원하기 위해 장면트리를 구성하는 객체들에 우선 순위를 부여하고 관리하고 있다. 현재 MPEG-4 표준에서 지원하고 있지 않은 사용자 상호작용을 지원하기 위해 상호작용을 분류하고 그 종류들을 정의하였다. 또한 상호작용이 발생시 중요한 문제인 동기화를 지원하기 위해 인과성 관계에 기반한 시간 관계 모델을 제안하고 있다. 사용자 상호작용이 발생되면 이 인과성 시간 관계에 기초하여 동기화 트리가 생성되어 동기화를 맞추게 된다.

향후, 연구 과제로서 MPEG-4에서 좀더 융통성 있는 상호작용을 지원하기 위해 XMT에 기반한 씬디스크립션에 대한 분석과, 이를 이용하여 다양한 멀티미디어 객체로 구성된 씬들을 기술하여 XMT 파일을 생성하고 재생하고 사용자 의도에 맞게 효과적으로 제어할 수 있는 시스템을 구현하는 것이다.

참 고 문 헌

[1] A. Puri, Multimedia System, Standards and Networks, Marcel Dekker, 2000.

[2] A.Puri and A Eleftheriadis, "MPEG-4 : An Object-based multimedia coding standard supporting mobil application," ACM J. Mobile Network Appl. Vol. 3, No. 5, pp,5-32, 1998.

[3] H. Radha et al., "Scalable Internet Video using MPEG-4," Signal Processing : Image Comm. Vol.15, pp.95-126, 1999.

[4] ISO/IEC 14496-1, Information Technology Coding of Audo-visual Object - Part 1 : System, ISO/IECJT/SC29/WG11, 1999, 12.

[5] R. Koenen, "MPEG-4 Overview-(V.21-Jeju Version)," ISO/IECJTC1/SC29/WG11N4668, March 2001.

[6] http://www.mpeg-4player.com/player.html

[7] http://drogo/cselt.it/mpeg/

[8] http://vdomail.net:8000

[9] S.Worrall et al., "Motion Adaptive Error Resilient Encoding for MPEG-4" IEEE Proc. of ICASSP, May 2001.

[10] Ralf Steinmetz "Synchronization Properties in Multimedia Systems," IEEE Journal on Selected Areas in Comm., Vol. 8, No. 3, Apr., 1990, pp. 401-412.

[11]Causal-Relation, http://www.darmstadt.gmd de/publish/komet/gen-um/node148-152.html.

최 숙 영 (Sook-Young Choi)

1988년 6월 전북대학교 이학사 (전산학).

1991년 2월 전북대학교 이학 석사 (전산학).

1996년 2월 충남대학교 이학 박사(전산학)

1996년 3월 ~ 현재 우석대학교 컴퓨터교육과 조교수

관심분야 : 멀티미디어 응용, 병렬처리, 원격교육