

MPEG-4 저작 시스템에서 BIFS 생성 모듈

(A BIFS Generation Module in an MPEG-4 Authoring System)

배수영^{*} 김상욱^{**} 마평수^{***}
(Suyoung Bae) (SangWook Kim) (Pyeongsoo Mah)

요약 기존의 MPEG-4 저작 도구들은 대부분 BIFS 전문가용 시스템으로 저작 도구 사용에 앞서 MPEG-4 콘텐츠 표현 구조인 BIFS에 대한 선지식을 요구하며, 저작자들은 이를 습득하기 위해 많은 시간과 노력을 소비해야 한다. 본 논문에서는 MPEG-4 콘텐츠 개발에 소요되는 이러한 노력을 절감하기 위해 사용자에게는 기존 멀티미디어 저작도구의 편리한 저작 환경을 제공하고, 이를 저작 도구 내부에서 BIFS로 변환하는 방법을 제시한다.

저작 환경은 BIFS 생성을 위해 요구되는 최소한의 저작 정보를 생성시키고, BIFS 생성 모듈은 이를 미리 정의해둔 BIFS 구성 규칙을 이용하여 BIFS로 재구성한다. 생성된 BIFS는 장면 트리, 객체 기술자, 연결경로(Route)로 구성되며, 이들은 MPEG-4 파일 포맷에 따라 기본 스트림과 먹싱되어 MP4 파일로 만들어진다.

키워드 : MPEG-4, BIFS, 멀티미디어 저작

Abstract Though BIFS is a complex description language that represents MPEG-4 content as text or bit streams, most traditional MPEG-4 content authoring tools are developed for a BIFS expert. Before using these tools, many authors must invest much time and energy in acquiring knowledge of BIFS. In this paper, we suggest a new manner for saving these efforts. Our proposal supplies the convenient user interface of traditional multimedia authoring tools for MPEG-4 authors and interprets the authored information and transforms it into BIFS.

In our user interface, the author generates the minimal authoring information requested for generating BIFS; the BIFS generation module then transforms the information into BIFS format using predefined BIFS construction rules. The resulting BIFS consists of the Scene Tree and the Object Descriptor and Route; they are used for constructing MP4 standard files by mixing with the Elementary Stream.

Key words : MPEG-4, BIFS, Multimedia Authoring

1. 서론

MPEG-4 콘텐츠 저작 시스템은 시청각 미디어 객체를 조합하여 사용자와 상호 작용하는 MPEG-4 콘텐츠를 생성하는 도구로서, MPEG-4에서 제시된 시청각 장면을 표현하기 위해 표준화된 형식인 BIFS(Binary Format for Scenes)[1]를 사용한다. MPEG-4 part1.

Systems[1]에 포함된 BIFS는 기본 개념이 VRML[2]로부터 나왔으며, 시청각 장면의 구조, 미디어 객체 합성, 변형, 전송에 관한 모든 정보를 표현하기 위해 제안된 것이다[1]. 하지만, BIFS는 그 형식과 내용 구조가 복잡하여 이를 직접 생성하거나 조작하기 위해서는 저작자의 많은 노력이 요구되며, 본 논문에서는 MPEG-4 콘텐츠 개발을 위해 소비되는 이러한 노력을 절감하기 위해 사용자에게는 편리한 저작 환경을 제공하고, 이를 내부적으로 MPEG-4 BIFS 형식으로 변환하는 모듈을 제시한다.

현재까지 개발된 MPEG-4 저작 도구로 MPEG-4 PC 프로젝트로 개발된 MDS(MPEG-4 Development Studio)[3]와 프랑스의 ENST에서 개발한 Harmonia[4]

^{*} 정 회 원 : 한국전자통신연구원 정보가전연구부 연구원
manim75@etri.re.kr

^{**} 비 회 원 : 경북대학교 컴퓨터학과 교수
swkim@cs.knu.ac.kr

^{***} 비 회 원 : 한국전자통신연구원 정보가전연구부 연구원
pmah@etri.re.kr

논문접수 : 2002년 1월 28일

심사완료 : 2002년 5월 31일

와 MPEG Pro[5-7], 그리스의 ITI에서 개발한 3D MPEG-4 저작 도구[8], envivio의 EBS[9] 등이 있다. MDS[3]와 MPEG Pro[5-7], ITI의 3D MPEG-4 저작 도구[8]는 각기 2차원, 3차원 MPEG-4 콘텐츠 저작 시스템으로 모두 전문가용 시스템이며, BIFS를 정확히 아는 사람만이 사용할 수 있다는 단점이 있다. Harmonia [4]는 MPEG-4 저작의 전문성을 탈피하기 위해 저작자에게 미리 생성된 템플릿을 제공하고, 저작자는 이를 자신에게 맞게 변경하도록 하는 편리한 구조로 되어 있으나, 동영상, 이미지, 텍스트, 슬라이드에 대해서만 템플릿이 개발되어 있다. EBS[9]는 사용자에게 BIFS 구조에 대한 선지식은 요구하지 않으나, 콘텐츠 구성에 필요한 모든 객체를 다른 도구를 통해 생성한 후, 이를 하나의 MPEG-4 콘텐츠로 조합하기 위한 환경만을 제공하므로, 저작자는 콘텐츠 생성을 위해 2 단계를 거쳐야 한다. 이외에 Philips[10]와 MPEG4IP[11] 등에서 MPEG-4 저작에 관해 연구 중이나 이들은 모두 MPEG-4 인코딩에 초점을 두고 있다.

따라서, 다양한 특징을 가진 MPEG-4 콘텐츠 저작을 쉽고 용이하게 하기 위한 연구가 필요하며, 본 논문은 MPEG-4 콘텐츠 저작을 위해 저작자에게 기존의 멀티미디어 저작의 편리한 환경[12]을 제공하고, 이를 내부적으로 MPEG-4 BIFS 형식으로 변환하는 방법을 제안한다. 본 논문의 제2장에서는 MPEG-4 저작 시스템 구조를 설명하고, 제3장에서는 MPEG-4 BIFS를 생성하는 모듈을 제시한다. 제4장에서는 다른 저작 시스템과의 비교 및 구현 결과를 보이고, 제5장에서 결론을 맺는다.

2. MPEG-4 콘텐츠 저작 시스템(MCAS: MPEG-4 Content Authoring System)

MPEG-4 콘텐츠 저작 시스템(MCAS:MPEG-4 Content Authoring System)은 복잡한 구조를 가진 MPEG-4 콘텐츠를 손쉽게 생성하는 방법을 제공한다. 저작자에게 기존 멀티미디어 저작 도구의 장점인 직관적인 저작 환경을 제공하고, 복잡한 BIFS 생성은 내부 생성 모듈을 사용해 은닉한다. 저작자가 저작한 내용은 내부의 BIFS 생성 모듈을 통해 재구성되며, 기본 스트림과 함께 지정된 파일 포맷에 따라 MPEG-4 콘텐츠로 완성된다. 그림 1은 MCAS의 구조를 나타낸다. MCAS는 사용자 저작 계층, BIFS 생성 계층, 파일 생성 계층의 3단계 계층을 가진다.

사용자 저작 계층은 저작자에게 직관적인 멀티미디어 저작 환경을 제공하고, 이 환경으로부터 저작된 데이터

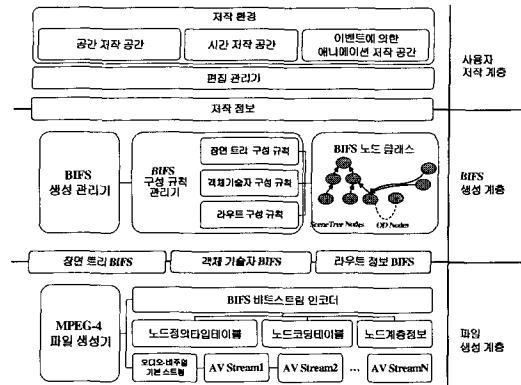


그림 1 MCAS의 구조

를 관리한다. 저작 환경은 공간 저작 공간과 시간 저작 공간, 이벤트에 따른 애니메이션 저작 공간으로 분리되며, 이들 공간에 저작되는 저작 객체(아이콘) 집합을 포함한다. 공간 저작 공간은 콘텐츠의 장면을 구성하기 위해 필요한 시각 미디어 객체를 나열하고, 오디오 객체에 대해 숨겨진 아이콘 형태로 첨가할 수 있는, 저작 도구의 메인 공간이다. 시간 저작 공간은 공간 작업 공간에 나열된 미디어 객체들에 대해 재생 시작과 종료 시간을 설정할 수 있는 윈도우이다. 이벤트에 따른 애니메이션 저작 공간은 각 객체에 대해 마우스와 시간 이벤트 발생 시 반응 액션, 즉 애니메이션을 설정하는 공간으로 애니메이션의 종류에 따라 몇 단계를 거치는 대화형 환경을 가진다. 저작된 내용은 편집 관리기를 통해 저작 정보에 저장된다.

BIFS 생성 계층은 저작 정보를 MPEG-4 콘텐츠로 재구성하기 위해 BIFS 생성 관리자, BIFS 구성 규칙, BIFS 노드 클래스를 가진다. BIFS 노드 클래스는 MPEG-4 Systems[1] 파트에 기록된 BIFS 노드 구조를 C++클래스에 맞게 구현한 것이며, BIFS 구성 규칙 관리기는 BIFS 노드 클래스를 이용, 장면 트리, 객체 기술자, 연결경로(Route)를 구성하기 위한 노드 조합 방법을 나열한 모듈이다. BIFS 생성 관리기는 사용자 저작 계층으로부터 전달된 저작 정보를 해석하여 BIFS 구성 규칙을 통해 BIFS 노드들로 재구성하고 이를 MPEG-4 파일 생성 계층에 전달한다.

파일 생성 계층은 BIFS 생성 계층에서 생성한 장면 기술, 객체 기술자, 연결경로 텍스트 스크립트를 MPEG-4 Systems[1]의 부록 H에 정의된 비트스트림으로 변환한다. 변환된 정보는 QuickTime[13] 포맷과 따르는 MPEG-4 파일 포맷[14]을 따라 기본 스트림과

혼합하여 하나의 파일로 만들어진다.

3. MPEG-4 저작 시스템에서 BIFS 생성 모듈

3.1 BIFS 생성 모듈의 입력 : 저작 정보

MCAS의 MPEG-4 BIFS 생성 모듈은 BIFS 생성에 필요한 최소한의 정보만을 저작 환경에서 제공받고, 이를 BIFS에 맞게 가공하여 사용한다. BIFS 생성에 필요한 저작 정보 AI(Authoring Information)는 다음과 같이 정의된다.

```
AI=<AVO_L, Grp_T, Ani_L>
AVO_L=<AVO_Id, IsGrp_flg>*
Grp_T=<Grp_Id, Grp_E_L>*
Grp_E_L= <Grp_E_Id, IsSubGrp_flg>*
Grp_E_Id = <AVO_Id|GrpO_Id>
Ani_L=<Ani_Id, Src_O, Dst_O, Evnt_T, Ani_T,
Key_V_L, Key_C >*
Evnt_T=<Time|OverTouch|ActiveTouch>
Ani_T=<Clr*|MvPnt*|Rtt*|ScI*>
```

저작 정보 AI는 자료의 효율적인 접근과 체계화를 위해 리스트와 테이블로 관리된다. AI는 시청각 객체 리스트 AVO_L과 그룹 객체 테이블 Grp_T, 애니메이션 정보 리스트 Ani_L를 가진다. AVO_L은 공간 저작 공간에서 편집된 기하 객체와 미디어 객체로 이루어지고, Grp_T는 공간 저작 공간에 나열된 객체들의 그룹 정보를 담고 있다. Ani_L은 이벤트에 따른 애니메이션 저작 공간에서 생성된 정보를 담고 있다.

AVO_L의 각 원소는 저작된 미디어 객체의 아이디인 시청각 객체 식별자 AVO_Id와 해당 시청각 객체가 그룹에 포함되는지 여부를 나타내는 플래그 IsGrp_flg로 구성된다. AVO_L에 포함될 수 있는 객체는 점, 선, 면, 텍스트를 포함하는 2차원 기하 객체와 이미지, 오디오, 비디오의 미디어 객체로서, 저작 도구는 각 객체별 그들의 모든 정보를 저장할 수 있는 C++ 클래스를 정의했다. 미디어 객체 저작 시, 해당 클래스의 인스턴스가 AVO_L에 첨가된다.

Grp_T는 그룹 식별자 Grp_Id와 해당 그룹에 속하는 구성 요소 리스트 Grp_E_L로 구성된다. Grp_E_L의 각 구성 원소는 해당 그룹에 포함된 객체의 식별자 Grp_E_Id와 그 객체가 그룹 객체인지 아닌지를 구분하기 위한 플래그 IsSubGrp_flg로 구성된다. Grp_E_Id는 시청각 객체 또는 다른 그룹 객체가 될 수 있으며, 따라서 식별자는 AVO_Id 또는 Grp_Id가 사용된다. 이들 그룹 정보는 MPEG-4 BIFS에서 장면 트리의 뼈대를 구성하는데 사용된다.

사용자의 마우스와 시간 이벤트에 대해 장면 변화의 내용을 담은 Ani_L은 애니메이션 정보 식별자 Ani_Id와 이벤트 발생을 인식하는 소스 객체 Src_O, 이벤트에 의해 애니메이션이 발생하는 목적지 객체 Dst_O, 발생되는 이벤트 종류 Evnt_T, 장면을 변화시키는 애니메이션 종류 Ani_T, 키프레임 값 리스트 Key_V_L, 키 프레임의 개수 Key_C로 구성된다. MPEG-4 콘텐츠에서 발생할 수 있는 이벤트 종류와 애니메이션 종류는 다음 표 1과 표 2에서 설명한다.

표 1 이벤트 종류

이벤트 형태	설명
Time	정의된 시간에 사용자가 지정한 애니메이션 수행
OverTouch	객체 위에 마우스가 위치할 때 해당하는 액션
ActiveTouch	객체가 선택될 때 해당하는 액션을 수행

표 2 애니메이션의 종류

애니메이션 종류	설명
Clr	객체의 색을 변경
MvPnt	객체를 이동
Rtt	객체를 회전
ScI	객체의 확대, 축소

3.2 BIFS 생성 모듈의 구성

BIFS 생성 모듈은 저작 정보 AI를 BIFS로 재표현한다. 정확한 BIFS 노드를 정의하고, 저작 정보를 BIFS로 표현하기 위해 미리 정의된 생성 규칙을 사용한다. 다음 정의는 BIFS 생성 계층 BIFS_GL을 구성하는 컴포넌트이다.

```
BIFS_GL = <BIFS_GM, BIFS_CRM,
BIFS_Nds_Cls>
BIFS_CRM = <ST_CRM, OD_CRM, R_CRM>
ST_CRM = <ST_GM, A_SM, Frm_CR,
MO_To_BIFS_Nds_CR>
OD_CRM = < OD_GM, OD_CR >
R_CRM = < R_GM, UE_R_CR, TE_R_CR >
```

BIFS 생성 계층 BIFS_GL은 BIFS 생성 관리기 BIFS_GM과 BIFS 구성 규칙 관리기 BIFS_CRM, BIFS 노드 클래스 BIFS_Nds_Cls로 구성된다. BIFS_GM은 저작 정보로부터 데이터를 읽어와서 장면 트리, 객체 기술자, 연결경로 생성을 주관한다. BIFS_CRM은 저작 정보를 BIFS 노드들로 표현하기 위해 미리 정의된 BIFS 노드 구성 규칙을 관리한다. BIFS_Nds_Cls는 MPEG-4 Systems[1]에 기록된 BIFS 노드들로

서, 속성 설정과 BIFS 텍스트 스크립트 출력 함수가 첨가된 C++ 클래스이다.

BIFS_CRM은 장면 기술 즉, 장면 트리 구성 규칙 관리기 ST_CRM과 객체 기술자 구성 규칙 관리기 OD_CRM, 연결경로 구성 규칙 관리기 R_CRM으로 구성된다.

ST_CRM은 시청각 객체들의 상하 계층 관계 및 그룹 관계를 표현하여 장면의 전체 구조를 결정하는 장면 트리 프레임 구성 규칙 Frm_CR과 장면 트리를 이루는 미디어 객체를 BIFS 노드들로 표현하는 객체별 BIFS 노드 구성 규칙 MO_To_BIFS_Nds_CR, 이들 구성 규칙을 토대로 BIFS 노드 클래스의 전체적인 조합하는 결정하는 장면 트리 생성 모듈 ST_GM, 생성된 장면 트리의 속성을 설정하는 속성 설정 모듈 A_SM로 구성된다.

OD_CRM은 객체 기술자 생성 규칙 OD_CR과 이를 바탕으로 객체 기술자를 구성하는 BIFS 노드들을 조합하는 객체 기술자 생성 모듈 OD_GM으로 이루어진다.

R_CRM은 저작 정보 중 이벤트에 따른 애니메이션을 표현하기 위한 모듈로서, 애니메이션을 수행하기 위해 필요한 노드를 장면 트리에 첨가하고, 각 노드, 필드 사이의 연결 관계를 스크립트로 표현한다. 연결경로 생성 규칙은 사용자 이벤트에 대한 연결경로 구성 규칙 UE_R_CR, 시간 이벤트에 대한 연결경로 구성 규칙 TE_R_CR 두 가지이며, 이들 구성 규칙은 연결경로 생성 모듈 R_GM을 통해 제어된다.

다음 그림 2는 저작정보와 BIFS 생성 계층을 구성하는 컴포넌트들의 상호작용을 정리한 것이다. 이들 컴포넌트는 장면 트리 생성시 필요한 객체 기술자를 동시에

생성하고, 그 후 연결경로를 생성하는 순서로 진행된다.

사용자가 저작 환경에서 MPEG-4 콘텐츠 생성을 요구하면, BIFS_GM은 장면 트리를 생성하기 위해 ST_CRM의 ST_GM을 호출한다. ST_GM은 먼저 장면 트리 프레임을 생성하기 위해 Frm_CR(1)을 이용하여 장면 트리를 구성하는 객체들의 그룹 관계를 BIFS 노드로 표현하고, MO_To_BIFS_Nds_CR(2)를 이용하여 장면 트리를 각 객체별 세부 정보까지 저장할 수 있도록 확장한다. ST_GM 동작 중 객체 기술자를 생성해야 하는 이미지, 비디오, 오디오 객체가 발견되면, OD_GM(3)이 호출되어 객체 기술자를 생성한다. (1)과 (2)를 통해 장면 트리의 전체 구조가 결정되면, A_SM(4)을 호출하여 장면 트리를 구성하는 노드들의 필드 값을 설정한다. 저작 정보에서 가져온 시청각 객체의 속성은 BIFS의 속성 표현법과 일치하지 않으므로, A_SM은 장면 트리에서 속성이 들어가는 필드를 검색하고, 속성 값을 BIFS 표현법에 맞게 변환하는 작업을 수행한다. R_CRM의 R_GM(5)은 이벤트에 따른 애니메이션 저작 공간에서 생성된 정보를 해석하여 UE_R_CR, TE_R_CR에 따라 장면 트리에 필요한 BIFS 노드를 첨가하고, 애니메이션 실행을 위해 노드 간 변경 정보 전달 경로를 나타내는 스크립트를 생성한다.

3.2.1 장면 트리의 뼈대 생성

장면 트리 생성 모듈 ST_GM은 장면 트리 프레임 생성 규칙 Frm_CR과 저작 정보 AI의 그룹 테이블 Grp_T, 시청각 객체 리스트 AVO_L를 이용해서 객체 간의 그룹 관계 및 상하 계층 관계를 표현하여 장면 트리의 기본 골격을 생성한다. 그림 3은 장면 트리 프레임 생성 알고리즘이다.

1 단계에서는 장면 트리를 구성할 저작 정보가 있는지 확인한 후, 장면 트리의 루트 노드를 생성한다. 장면 트리의 루트 노드는 Group 노드가 된다.

2 단계에서는 저작 정보 AI의 그룹 테이블 Grp_T를 읽어들이 각각의 그룹을 BIFS 노드로 표현하며, 표현된 그룹은 그룹 리스트 GL에 저장한다. GL의 각 요소는 그룹 객체 핸들 grpobj와 현재 그룹이 다른 그룹의 하위 그룹인지를 나타내는 플래그 sub_flg로 구성된다. sub_flg 값은 현 단계에선 기본 값으로 설정되어 있으며, 3 단계에서 갱신된다. 그룹 관계 테이블 GR은 GL의 각 그룹이 포함하는 하위 그룹을 저장하기 위한 것으로, 하위 그룹의 개수와 아이디를 갖고 있다.

3 단계에서는 GL의 각 그룹에 대해 GR에 명시된 값, 즉 하위 그룹들을 조사, 정리하여 그룹간의 상하 관계를 표현한다. GL의 각 그룹을 인덱싱하면서, 현재 지시하

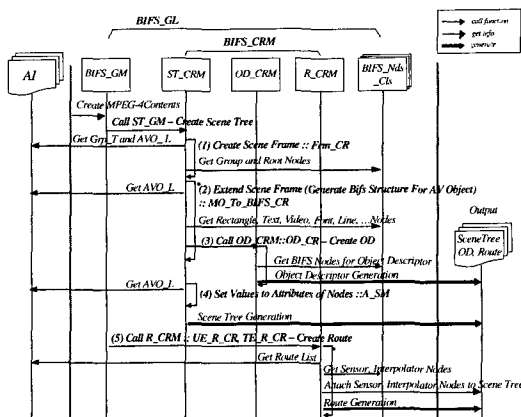


그림 2 BIFS 생성 모듈의 컴포넌트 상호작용

```

1. AI 시작 정보 = < AVO_L, Grp_T, Ani_L >
2. 장면 트리 생성을 위해 필요한 내부 변수 및 자료구조.
  2.1 grp_cnt : 그룹 개수 카운터
  2.2 subgrp_cnt : 각 그룹에 대한 서브 그룹 카운터
  2.3 GR[i][j] : i 그룹의 서브 그룹 j의 핸들
  2.4 GL 그룹 리스트 = < grpobj, sub_flg >
  G.[i].grpobj : Ai의 Grp_T의 i번째 내용대로 구성된 BIFS 그룹
  G.[i].sub_flg : 현재 그룹의 다른 그룹 종속 유무를 나타내는 플래그

Step 1. 장면 트리의 루트 노드를 생성한다.
  If( AI->NumOfAVObj) exit(0); /* 시작 정보가 없을 경우 종료 */
  else Root = new Group(); /* 루트 노드 생성 */

Step 2. 시작 정보의 그룹 테이블의 읽어들이어 BIFS 노드로 이루어진 그룹을 생성한다.
  If( AI->NumOfGrp > 0 ){ /* 그룹이 존재할 경우 */
    grp_cnt = 0;

    while( the number of group ){
      subgrp_cnt = 0;
      Transform2D *InternalGrp = new Transform2D(); /* 그룹 객체 생성 */
      while( For all elements in group ){
        Crp_E *obj = AI->Grp_T[grp_cnt]->Grp_E_L_.GetNext();

        If( obj is group object ){
          /* 서브 그룹 발생: 그룹 관계 테이블에 저장 */
          GR[grp_cnt][subgrp_cnt] = obj->GrpNum;
          subgrp_cnt++;
        } else {
          /* 수평한 그룹 구성 요소가 그룹이 아닌 시청각 객체인 경우 */
          /* 시청각 객체에 대응하는 Transform2D를 생성하고, Transform2D에
          해당 시청각 객체 핸들을 설정한 후, 그룹 객체에 연결 */
          Transform2D *InternalAVO = new Transform2D();
          InternalAVO->SetHandle(obj->Handle);
          InternalGrp->Attach( InternalAVO );
        }
      }
      GL[grp_cnt].grpobj = InternalGrp;
      GL[grp_cnt].sub_flg = false;
      grp_cnt++;
    }
  }
  
```

```

Step 3. 그룹 관계 테이블에서 그룹 사이의 포함 관계를 읽어들이어, 그룹 사이
종속관계를 생성한다.
  grp_cnt = 0;
  while( the number of group ){
    subgrp_cnt = 0;
    while( the number of subgrp in current group ){
      /* 그룹 리스트의 현재 그룹에서, 그룹 관계 GR에 명시된 현재 그룹의
      하위 그룹을 현재 그룹에게 연결 */
      GL[grp_cnt].grpobj->Attach(GL[subgrp_cnt][subgrp_cnt].grpobj);
      GL[GR[grp_cnt][subgrp_cnt]].sub_flg = true;
      subgrp_cnt++;
    }
    grp_cnt++;
  }

Step 4. 장면 트리의 루트 노드에 그룹 노드와 그룹에 포함되지 않는 시청각
객체를 표현한 노드를 추가한다.
  grp_cnt = 0;

  while( the number of group ){
    /* 다른 그룹에 종속되지 않은 그룹을 장면 루트에 연결 */
    If( GL[grp_cnt].sub_flg != true ){
      Root->Attach( GL[grp_cnt].grpobj );
      grp_cnt++;
    }
  }

  while( for all AV Object ){
    /* 그룹에 포함되지 않는 객체를 장면 루트에 연결 */
    /* 그룹에 포함되지 않는 시청각 객체에 대응하는 Transform2D를 생성하고,
    Transform2D에 해당 시청각 객체의 핸들을 설정한 후, 장면 트리에 연결 */
    If( this AV Object is not included in groups ){
      Transform2D *InternalAVO = new Transform2D();
      InternalAVO->SetHandle( obj->Handle );
      Root->Attach( InternalAVO );
    }
  }
  
```

그림 3 네 단계를 가진 장면 트리 프레임 생성 알고리즘

는 그룹이 GR에서 하위 그룹을 가질 경우, 해당 하위 그룹을 현재 그룹에 첨가시킨다.

4 단계에서는 다른 그룹에 종속되지 않는 그룹을 가려내고, 이를 장면 트리의 루트에 연결해서 장면 트리 프레임을 완성한다. 그룹에 포함되지 않는 객체 또한 독립적인 그룹을 이루는 객체로 간주하여 장면 트리 루트에 연결한다.

이들 과정은 다음 그림 4에 요약된다.

3.2.2 장면 트리 확장

장면 트리 프레임이 결정되면, 장면 트리 생성 모듈

ST_GM은 미디어 객체에 대한 BIFS 노드 구성 규칙 MO_To_BIFS_Nds_CR를 이용하여 장면 트리 뼈대의 말단을 형성하는 각 시청각 객체를 BIFS 노드들로 확장한다. MO_To_BIFS_Nds_CR은 세 종류로 구분되며, 각각에 대한 생성 단계는 표 3과 같다.

MO_To_BIFS_Nds_CR은 2차원 기하 객체 생성 규칙과 비주얼 스트림 객체 생성 규칙, 오디오 스트림 객체 생성 규칙으로 나뉜다. 기하 객체 생성 규칙을 따르는 객체는 점, 선, 사각형, 원, 텍스트 등이 있으며, 비주얼 스트림 객체로는 이미지와 비디오가 있다.

2차원 기하 객체와 비주얼 스트림 객체는 'Shape' 노드를 기본으로 해서 객체의 외형을 채우는 'Appearance' 노드와 객체의 고유한 형태를 나타내는 노드들로 구성된다. 기하 객체의 경우 'Appearance' 노드에 객체의 색과 선, 객체 내부의 필링 유무를 나타내기 위한 속성이 들어있고, 객체의 고유 형태를 나타내는 노드로 'Rectangle', 'Text', 'LineSet2D', 'PointSet2D', 'Circle' 등이 사용된다. 비주얼 스트림 객체의 경우, 두 객체 모두 타임 스탬프에 따른 표현 양식은 이미지이므로, 객체의 고유한 형태를 나타내는 노드는 'Bitmap' 형태로 동일하지만, 'Appearance'에 'ImageTexture'와 'MovieTexture'를 각기 첨가함으로써 어떤 내용이

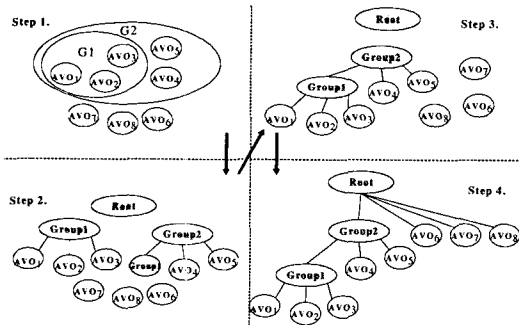


그림 4 장면 트리 뼈대 생성 과정

표 3 장면 트리 확장을 위한 각 객체별 BIFS 노드 생성 규칙

시청각 객체	생성 규칙
2D Geometry Objects	Step 0. Create Shape, Appearance, Material2D, LineProperties Nodes Step 1. Attach LineProperties To Material2D Step 2. Attach Material2D To Appearance Step 3. Attach Appearance To Shape Step 4. if (AVObj.type == Text) then Create Text, FontStyle Nodes Attach FontStyleNode To Text if (AVObj.type == Rectangle) then Create Rectangle Node if (AVObj.type == Circle) then Create Circle Node if (AVObj.type == Line) then Create Coordinate2D, IndexedLineSet2D Nodes Attach Coordinate2D to IndexedLineSet2D if (AVObj.type == Point) then Create Color, Coordinate2D, PointSet2D Nodes Attach Color and Coordinate2D to PointSet2D Step 5. Attach Geometry Node(Text, Circle, IndexedLineset2D or PointSet2D) to Shape
Visual Stream Objects	Step 0. Create Material2D, Appearance, Bitmap, Shape Nodes Step 1. Attach Material2D To Appearance Step 2. if(AVObj.type == Image) then Create ImageTexture Node if (AVObj.type == Video) then Create MovieTexture Node Step 3. Attach Visual Stream Node(ImageTexture or MovieTexture) To Appearance Step 3. Attach Bitmap To Shape Step 4. Attach Appearance To Shape Step 5. Call GenerateOD method in OD_CRM
Audio Stream Objects	Step 0. Create AudioSource, Sound2D Nodes Step 1. Attach AudioSource To Sound2D Step 2. Call GenerateOD method in OD_CRM

'Bitmap'으로 표현되는지 구별하게 한다. 오디오 스트림 객체와 비디오 스트림 객체는 각각에 대해 객체 기술자를 생성해야 하므로 객체 기술자 구성 규칙 관리자 OD_CRM을 호출하는 모듈을 포함한다.

3.3 객체 기술자 생성

MPEG-4 Part.1 시스템에 기술된 객체 기술자는 다음과 같이 구성된다.

```
OD = < OD_ID, ES_Descriptor >
ES_Descriptor = < ES_ID, DecoderConfig,
                SLConfig, MuxInfo >
```

객체 기술자 생성 규칙 OD_CR은 OD를 생성해야 하는 시청각 객체에 대한 핸들을 획득한 후, OD 노드와 ES_Descriptor 노드를 생성하고, ES_Descriptor에 포함되어야 할 MuxInfo, DecoderConfig, SLConfig 노드들을 생성한다. MuxInfo에는 현재 객체 기술자가 나타내는 스트림의 파일 시스템 내의 위치와 타입을 설정하고, DecoderConfig는 스트림이 적당한 디코더를 통해 디코딩 되어 사용자에게 보여지도록 스트림의 타입과 디코딩 버퍼 크기 정보를 설정한다. SLconfig에는 기본 스트림을 원하는 시점에서 읽고 디코딩해서 다른 스트림과 동시에 맞게 재생시킬 수 있도록 타임스탬프 값을 설정한다. 세 가지 노드에 대해 필요한 값이 설정되면,

ES_Descriptor가 이들의 핸들을 소유하도록 한다. 필요한 모든 노드를 가지게 된 ES_Descriptor에 식별자 ES_ID를 할당하고, 이를 미리 생성한 OD와 연결한다. OD의 내용이 완성되었으므로 OD의 식별자 OD_ID를 생성한다.

3.4 연결경로 생성

MPEG-4 콘텐츠 저작자는 사용자 입력과 시간 변화에 따라 객체의 이동이나 변형을 정의하여 사용자에게 동적인 장면을 제공한다. 객체의 이동이나 변형을 자연스럽게 표현하기 위해서 MPEG-4 BIFS는 키프레임 애니메이션 기법을 사용하며, 이를 위해 색 변경[Color-Interpolator], 좌표계 위치 변경[CoordinateInterpolator], 객체 이동[PositionInterpolator], 확대/축소[Scalar Interpolator], 회전[OrientationInterpolator] 등의 인터플레이터 노드가 정의되어 있다. 각 인터플레이터 노드는 키프레임의 재생 시간을 나타내는 Key와 그 때의 키프레임 정보 Key Value를 소유하고 있다.

연결경로는 애니메이션 실행을 위해 이벤트를 입력 받는 노드와 인터플레이터 노드 사이의 연동 정보를 나타내는 것으로, 연결경로 구성 규칙 관리자 R_CRM는 저작 정보 AI의 애니메이션 정보 리스트 Ani_L을 바탕으로 연결경로를 생성한다. 다음은 생성 단계를 나타낸다.

첫 단계는 Ani_L의 이벤트 발생을 인식하는 소스 객체 Src_O, 이벤트에 의해 애니메이션이 발생하는 목적지 객체 Dst_O에 대응하는 장면 트리의 노드 핸들 Src_Nd, Dst_Nd를 가져온다. 이들 핸들은 연결경로를 생성하는 과정에서 새롭게 추가되는 노드들 즉, 키프레임 애니메이션을 위한 시간 값 관리 노드 TimeSensor, 사용자 마우스 이벤트를 입력받는 노드 TouchSensor, 앞서 언급한 인터플레이터 노드를 장면 트리에 첨가시키는데 이용된다.

두 번째 단계는 TouchSensor, TimeSensor를 생성하여 장면 트리에 연결한다. TouchSensor는 이벤트 수신 노드인 Src_Nd에 첨가하고, TimSensor 노드는 애니메이션이 발생하는 노드 Dst_Nd에 첨가한다.

세 번째 단계는 Ani_L의 애니메이션 종류 Ani_T에 따라 인터플레이터 노드를 생성하고, 이를 Dst_Nd의 하위 노드로 첨가한다. 생성된 인터플레이터 노드에 Ani_L에 명시된 키 프레임의 개수 Key_C 만큼 Key를 생성하고, Ani_L의 키 프레임 값 리스트 Key_V_L로 인터플레이터 노드의 Key Value를 채운다.

마지막 단계는 생성된 TouchSensor, TimeSensor, 인터플레이터 노드의 식별자를 이용하여 연결경로 텍스트 스크립트를 생성한다.

위의 연결경로 생성은 사용자 마우스 입력에 따른 것

을 기준으로 만들어진 것이며, 시간의 변화에 따른 장면의 자동 변화는 각 단계에서 TouchSensor와 관련된 부분이 삭제된다.

4. 구현 및 결과

본 논문에서는 일반 멀티미디어 저작자가 편리한 저작 환경을 이용하여 쉽게 MPEG-4 콘텐츠를 생성할 수 있도록 하기 위해, 저장된 정보를 내부 모듈을 통해 복잡한 BIFS로 재구성하는 방법을 제시하였다. 표 4는 본 논문에서 제시한 시스템 MCAS가 복잡한 BIFS 생성을 내부로 은닉함으로써 사용자에게 제공하게 되는 저작 환경 및 저작 방식과 각 시스템의 BIFS 생성 방식에 대한 비교를 보인다.

표 4에 나타난 모든 저작 시스템은 뷰 사이의 동기화 [15]를 제공함으로써 저작자들에게 멀티미디어 콘텐츠 저작에 대한 능률을 높여주었으나, 각각은 MCAS에 비해 다음과 같은 단점을 가진다.

MDS[3], MPEG-Pro[5-7], 3차원 MPEG-4 저작 도구[8]는 저작 환경의 장면 트리 뷰를 통해 사용자에게 BIFS의 구조를 직접 보여주고, 사용자는 이벤트에 따른 액션 저작을 장면 트리에 바로 삽입하는 BIFS 전문가용 시스템들로서, 일반 사용자가 사용하기 위해서는 BIFS 구조에 관한 연구가 필요하다. 또한, 객체들의 제

표 4 기존 시스템과의 비교

	MCAS	MDS	MPEG-Pro	3차원 MPEG-4 저작 도구	Harmonia	EBS
저작내용	complete 2D Scene Graph profile 지원 (AnimaitonStream, AudioFX/Mix/Delay, TermCap노드 제외)	이미지, 2차원 기하 객체	complete2D Scenc Graph profic	complete3D Scenc Graph profile	슬라이드 쇼, 동영상, 이미지, 버튼, 텍스트	비디오, 오디오, 이미지, 플래쉬로 작성된 문자와 기하 객체
객체생성방식	직간접	직간접	직간접	직간접	직간접	간접
저작환경	편집 뷰 + 애니메이션 뷰 + 다중시간 뷰	편집 뷰 + 장면 트리 뷰 + 통합시간 뷰	편집 뷰 + 장면 트리 뷰 + 통합시간 뷰	편집 뷰 + 장면 트리 뷰	편집 뷰 + 객체별 편집 도구	편집 뷰 + 애니메이션 뷰 + 다중시간 뷰
저작방식	공간저작	WYSIWYG	제한된 WYSIWYG	제한된 WYSIWYG	제한된 WYSIWYG	제한된 WYSIWYG
	시간저작	다중시간 윈도우 객체별 시각 저작	없음	스크립트 방식 + 내부 자동 갱신	없음	다중시간 윈도우 객체별 시각 저작
	이벤트에 따른 액션저작	가능한 설정을 단계별로 제시하는 대화형 환경	장면 트리 뷰에 필요한 노드 직접 첨가	장면 트리 뷰에 필요한 노드 직접 첨가	장면 트리 뷰에 필요한 노드 직접 첨가	움직임 설정 공간 사용(일부 이벤트 지원)
BIFS 생성	2단계 확장 방식을 이용한 BIFS 자동 생성	직접 조작 + 내부 모듈 통한 BIFS 생성	직접 조작 + 내부 모듈 통한 BIFS 생성	직접 조작 + 내부 모듈 통한 BIFS 생성	템플릿을 이용한 BIFS 자동 생성	저작 객체의 포맷 변환을 통한 BIFS 생성
사용계층	일반 멀티미디어 저작자 및 초보자	BIFS 전문가	BIFS 전문가	BIFS 전문가	초보자	일반 멀티미디어 저작자 및 초보자

생 시간 저작을 속성이나 스크립트로로 지정해야 하기 때문에, 사용자는 객체들의 재생 시간에 관한 정보를 모두 기억하고 있어야만 한다.

Harmonia[4]는 MPEG-Pro[5-7]에서 저작되는 내용을 추상화시킨 것으로, 각 객체를 표현하는 BIFS의 구조를 템플릿으로 사용자에게 제공하고, 사용자는 제공된 템플릿의 속성을 자신에게 맞게 설정하여 함으로서 MPEG-4 콘텐츠를 저작한다. 하지만, 지원되는 템플릿의 종류가 동영상, 이미지, 텍스트, 버튼, 슬라이드 쇼로 제한되어 있고, 이벤트에 대한 액션 또한 몇 가지(시작, 멈춤, 보여주기, 숨기기, 링크)로 한정되어 있어, 사용자는 다양한 특성을 가진 콘텐츠를 생성할 수 없다.

EBS[9]는 일반 멀티미디어 사용자와 초보자를 위해 BIFS 구조를 내부에 숨기고, 객체들의 시간 관계를 시각적으로 저작하는 편리함을 갖고 있다. 그러나, 다양한 저작 도구를 통해 나오는 결과물의 조합만으로 MPEG-4 콘텐츠를 재구성하는 간접적인 객체 생성 방식을 택하고 있기 때문에, 텍스트와 기타 객체 등의 간단한 객체에 대해서 수정하고자 할 때에도 해당 객체를 생성한 도구를 다시 거쳐야 하는 불편함을 가진다.

MCAS는 MPEG-Pro[5-7]에 비해 제한된 complete 2D 프로파일을 제공하지만, 다양한 2차원 객체에 대해 직간접적인 생성 방식을 제공하고, 다중시간선 뷰와 애니메이션 뷰, BIFS 자동 생성 모듈을 통하여 일반 멀티

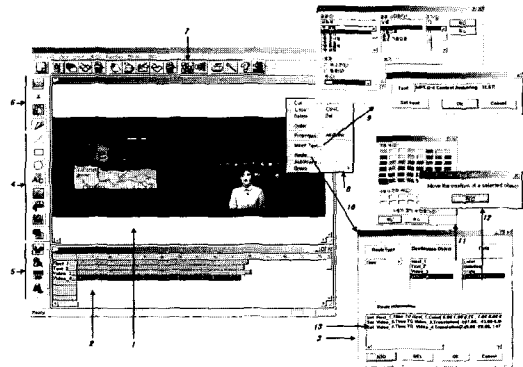


그림 5 MCAS의 저작 환경과 콘텐츠 저작

미디어 저작자도 쉽게 MPEG-4 콘텐츠를 저작할 수 있도록 한다. 그림 5는 MCAS에서 간단한 콘텐츠를 저작하는 모습을 보인다.

저작 환경은 공간 저작 공간(1)과 시간 저작 공간(2), 이벤트에 따른 애니메이션 저작 공간(3)과 아이콘 집합으로 구성된다. 아이콘 집합은 저작 객체 아이콘(4), 객체별 속성 지정 아이콘(5), 편집 아이콘(6), 파일 생성과 미리보기 아이콘(7)으로 구성된다.

공간 저작 공간(1)에 저작된 콘텐츠는 사각형과 텍스트, 두 개의 H.263 비디오로 구성되어 있으며, 이들에 대한 시간 정보가 시간 저작 공간(2)에 설정되어 있다.

```

Group {
  children [
    Scene Tree
    DEF Trans1 Transform2D {
      translation 15.00 -12.50
      scale 1.00 1.00
      rotationAngle 0.00
      children [
        Shape {
          appearance Appearance {
            material DEF Material2 Material2D {
              emissiveColor 0.00 0.00 0.00
              filled TRUE
              lineProps LineProperties {
                transparency -1.00
              }
            }
          }
          geometry Rectangle {
            size 884.00 225.00
          }
        }
        DEF TimeS0 TimeSensor {
          cycleInterval 3.000000
          enabled TRUE
          loop TRUE
          startTime 0.000000
          stopTime -1.000000
        }
        DEF ColorInter1 ColorInterpolator {
          key [ 0.00 0.33 0.67 1.00 ]
          keyValue [
            0.00 1.00 0.25 1.00 0.00 0.00
            0.00 1.00 1.00 0.50 0.00 1.00
          ]
        }
        DEF Trans2 Transform2D {
          translation -13.00 160.00
          scale 1.00 1.00
          rotationAngle 0.00
          children [
            Shape {
              appearance Appearance {
                material Material2D {
                  emissiveColor 1.00 1.00 0.00
                  filled TRUE
                  lineProps LineProperties {
                    transparency -1.00
                  }
                }
              }
            }
          ]
        }
      ]
    }
    geometry Text {
      string ["MPEG-4 Content Authoring : TEST"]
      fontStyle FontStyle {
        horizontal FALSE
        size -27.00
        style "PLAIN"
      }
    }
    DEF Trans3 Transform2D {
      translation -225.00 -30.00
      scale 1.00 1.00
      rotationAngle 0.00
      children [
        Shape {
          appearance Appearance {
            material Material2D {
              emissiveColor -1.00 -1.00 -1.00
              filled FALSE
              transparency -1.00
            }
          }
          texture MovieTexture {
            loop FALSE
            speed 1.00
            startTime 1.00
            stopTime 10.00
            url |
            repeats TRUE
            repeat TRUE
          }
          geometry Bitmap {
          }
        }
        DEF TimeS3 TimeSensor {
          cycleInterval 3.000000
          enabled TRUE
          loop TRUE
          startTime 0.000000
          stopTime -1.000000
        }
        DEF PositionInter2D4 PositionInterpolator2D {
          keyValue [
            -207.00 -43.00 -8.00 24.00
            95.00 -138.00 95.00 -138.00
          ]
        }
      ]
    }
  ]
}
Route
ROUTE TimeS0.fraction_changed
TO ColorInter1.set_fraction
ROUTE ColorInter1.value_changed
TO Material2.emissiveColor

ROUTE TimeS3.fraction_changed
TO PositionInter2D4.set_fraction
ROUTE PositionInter2D4.value_changed
TO Trans3.translation

ROUTE TimeS5.fraction_changed
TO PositionInter2D6.set_fraction
ROUTE PositionInter2D6.value_changed
TO Trans4.translation

Object Descriptor
{
  objectDescriptorID 1
  esDescr
  {
    .ES_ID 200
    muxInfo {
      fileName "C:\WConnect\h263"
      streamFormat H263
    }
    decConfigDescr {
      objectTypeIndication 0xC2
      streamType 4
      bufferSize 200000
      decSpecificInfo
      DecoderSpecificInfoString {
        info "obsolete string"
      }
    }
    siConfigDescr {
      useAccessUnitStartFlag TRUE
      useAccessUnitEndFlag TRUE
      useRandomAccessPointFlag TRUE
      useTimeStampsFlag TRUE
      timeStampResolution 1000
      timeStampLength 10
      AU_seqNumLength 8
      packetSeqNumLength 3
    }
  }
}
  
```

그림 6 BIFS 생성 모듈을 통해 생성된 장면 트리와 객체 기술자, 연결경로

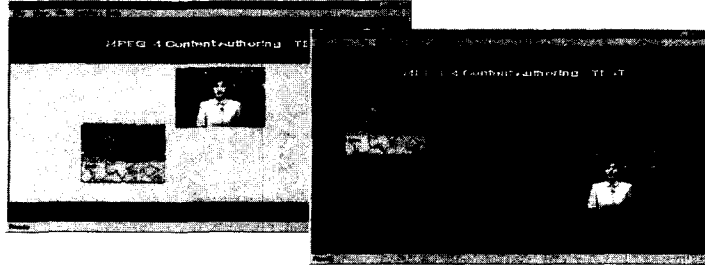


그림 7 IM-2D 재생기를 통한 재생

저작자는 저작된 객체 상에서 오른쪽 버튼 클릭으로 나타나는 팝업 상자(8)를 통해 객체의 이벤트에 따른 애니메이션 저작 공간(3)을 호출하고, 상자가 나타나어 주는 4 단계를 밟아 애니메이션을 저작한다. 저작 화면에서 사각형은 사용자 마우스 입력에 따른 색 변화가 설정되고(11), 두 개의 비디오는 시간이 지남에 따른 객체 위치 변경 액션(12)이 설정되어 있다. 설정된 내용은 스크립트로 자동 변환되어 저장(13)되고, 저작 환경에서 생성된 모든 정보는 내부의 BIFS 생성 모듈을 통해 다음 6과 같이 재구성된다.

그림 6의 장면 트리는 사각형과 텍스트, 첫 번째 비디오의 BIFS 구성을 보이고, 연결경로에서는 시간별로 사각형의 색 변화와 비디오의 위치 변환을 위해 장면 트리에서 어떤 노드들이 연결되어야 하는지 알려준다. 객체 디스크립터는 첫 번째 비디오가 어떤 타입으로 어느 위치에 저장되어 있는지 MP4 파일 생성 계층에서 사용할 먹싱 정보와, 디코딩과 전송을 위해 필요한 환경 설정 정보가 들어 있다. 이렇게 구성된 BIFS는 MCAS의 파일 생성 계층을 통해 다른 미디어 파일들과 혼합되어 하나의 MPEG-4 파일로 만들어지며, IM-2D 재생기를 통해 재생된다. 그림 7은 생성된 콘텐츠를 IM-2D 재생기를 통해 재생했을 경우, 시간의 변화에 따른 장면이 변화되는 내용을 나타낸다.

5. 결론

본 논문에서는 MPEG-4 콘텐츠 개발을 위해 소비되는 노력을 절감하기 위해 사용자에게는 편리한 저작 환경을 제공하고, 이를 저작 도구 내부에서 MPEG-4 BIFS로 변환하는 계층을 제시했다. BIFS 생성은 2차원 장면 구성으로 제한되었으며, 그 생성 포맷도 MPEG-4 표준 제정 그룹의 참조 소프트웨어인 IM-2D 재생기에 맞게 구성되었다. 앞으로, 좀 더 많은 기능 제공을 위해 지원되는 프로파일의 확장이 필요하며, 현재 고유의 포

맷으로 이루어지는 중간 저장을 MPEG-4 파일로부터 읽어들이 수정하도록 하는 작업과 저작 도구에서 스트리밍 서버로 생성된 데이터의 실시간 전송에 관한 연구가 진행 중이다.

참고 문헌

- [1] ISO/IEC FCD 14496-1 "Systems," 1999-12-15.
- [2] Andreal L. Ames, David R. Nadeau and John L. Moreland "VRML 2.0" 2nd Ed. WILEY.
- [3] MPEG-4 Tools by ENST : MPEG-4 Development Studio, <http://smil.nist.gov/IM1/player/enst/index.html>
- [4] Boughoufalah, S., Brelot, M., Bouilhaguet, F. and Dufourd, J., "A Template-guided authoring environment to produce MPEG-4 content for the web," Proceedings of MediaFutures2001. 2001.
- [5] Bouilhaguet, F, Concolato, C., Boughoufalah S. and Dufourd, J., "Adding Delivery support to MPEG-Pro, an Authoring System for MPEG-4," Packet Video'01. 2001.
- [6] Boughoufalah S, Dufourd J-C and Bouilhaguet F. "MPEG-Pro, an authoring system for MPEG-4 with temporal constraints and template guided editing," Proceedings of ICME2000, vol.1, pp.175-8, Piscataway, NJ, USA.
- [7] Souhila B., Jean-Claude D. and Frederic B. "MPEG-Pro, an Authoring System for MPEG-4," Proceedings of ISCAS 2000, May 28-31, 2000.
- [8] Petros D., Ioannis K., Theodoros R. and Michael G. S. "MPEG-4 authroing tool for the composition of 3D audiovisual scenes," Proceedings of ISCAS 2001, Vol 2., pp 201-4, May 6-9, 2001.
- [9] <http://www.envivio.com/>
- [10] <http://www.mpeg-4player.com/products/server/index.asp>
- [11] <http://mpeg4ip.sourceforge.net/>
- [12] Macromedia software, <http://www.macromedia.com/software/>
- [13] QuickTime FileFormat Specification, Apple Com-

puter Inc, May 1996.

- [14] ISO/IEC 14496-1/FDAM-1 "MPEG-4 version 2 Intermedia Format-MP4," 1999.11.
- [15] Muriel Jourdan, Cecile Roisin and Laurent Tardif "Multiviews Interfaces for Multimedia Authoring Environments," Proceedings 1998 MultiMedia Modeling, MMM'98. pp.72-9. Los Alamitos, CA, USA, 1998.



배수영

1998년 경북대학교 컴퓨터학과 졸업(학사). 2000년 경북대학교 대학원 컴퓨터학과 졸업(석사). 2000년 2월 ~ 2001년 4월 삼성전자 정보통신 연구원. 2001년 4월 ~ 현재 한국전자통신연구원 정보기전연구부 연구원. 관심분야는

MPEG, 멀티미디어, 미디어 저작 및 스트리밍

김상욱

정보과학회논문지 : 컴퓨팅의 실제
제 8 권 제 1 호 참조



마평수

1985년 서울대학교 식물병리학과 졸업(학사). 1992년 City University of New York, USA 전산학과 졸업(석사). 1995년 Wright State University, USA 전산학과 졸업(박사). 1985년 ~ 1989년 시스템공학연구소 연구원. 1989년 ~ 1990

년 (주)태양금속 정보산업연구실 대리. 1996년 ~ 현재 한국전자통신연구원 컴퓨터소프트웨어기술연구소 책임연구원. 관심분야는 멀티미디어 저장 서버, 지능정보단말