

MPEG-4 파일 포맷 분석을 통한 I-VOP추출 알고리즘 설계 및 구현

김용빈*, 김준일*, 신동일*, 신동규*, 박지현**, 이준석**

세종대학교 컴퓨터공학과

한국 전자통신 연구원 컴퓨터 소프트웨어 연구소**

e-mail: {kimybin, junil, dshin, shindk}@gce.sejong.ac.kr

{junhyun, junsuk}@etri.re.kr

Design and Implementation of the I-VOP extraction algorithm through analysis of the MPEG-4 file format

Yong-Bin Kim*, Jun-Il Kim*, Dong-Il Shin*, Dong-Kyoo Shin*,
Ji-hyun Park* Jun-suk Lee*

*Dept of Computer Science, Sejong University

Electronic and Telecommunications Research Institute**

요 약

MPEG-4는 “내용 기반 부호화”를 전제로 하는데, 영상을 객체 단위로 나누어 처리하므로, 사용자의 의도에 따라 다양한 형태의 조작과 디스플레이가 가능하다. 따라서 본 논문에서는 MPEG-4 파일의 분석을 통한 비주얼 요소중 가장 중요한 I-VOP을 추출하는 알고리즘을 설계 및 구현하여 보다 다양한 멀티미디어 서비스에 적용하는 방안을 마련하였다.

1. 서론

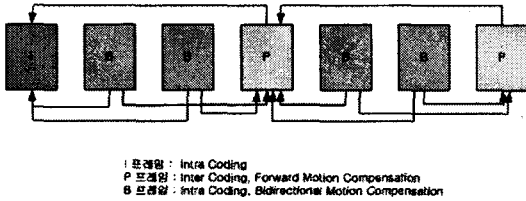
최근 멀티미디어 서비스에 대한 요구가 크게 늘어 남에 따라 멀티미디어 통신의 핵심기술인 초저속 영상/음성 신호 부호화 방식에 많은 관심이 모아지고 있다. 기존의 음성신호 뿐만 아니라, 영상 신호까지 포함한 방대한 정보량을 가지는 멀티미디어 신호를 유선 통신망이나 무선 채널을 통해 효율적으로 전송하기 위해서는 디지털 신호의 압축처리가 절대적으로 필요하다. 현재 ITU-T SG16산하의 LBC 그룹 (Experts Group for Very Low Bitrate Visual Telephony)과 ISO/IEC JTC1/SC29/WG11 의 MPEG-4(Moving Picture Expert Group 4)에서 이에 대한 표준화 작업을 활발히 진행하고 있다.[3] MPEG-4는 “내용 기반 부호화”를 전제로 하는데, 영상을 객체 단위로 나누어 처리하므로, 사용자의 의도에 따라 다양한 형태의 조작과 디스플레이가 가능하다.[1] MPEG-4는 무선이나 DSL망처럼 속도가 낮은 네트워크를 통해 전송하는 멀티미디어 응용 프로그램에 적용하는데 매우 적합한 표준이며 이동전

화, 비디오 전화, 개인 휴대정보단말기(PDA), 원격 영상회의, 인터넷, 양방향 멀티미디어 스트리밍, 양방향 방송, DVD, 캠코더 등 광범위한 부분에 적용된다.[7] 따라서 본 논문에서는 다양한 멀티미디어에 표준이 되고 있는 MPEG-4 파일 포맷을 분석하여 비주얼 요소중 가장 중요한 I-VOP 추출 알고리즘을 설계 및 구현하였다.

2. I-Frame 추출 적용사례

MPEG-4은 [그림 1]과 같이 I-VOP, B-VOP, P-VOP 이 혼합된 형태의 구조를 가지고 있다. 이 때 B-VOP과 P-VOP은 다른 프레임에 참조해야 복호화가 가능한 반면, I-VOP은 자신만으로 복호화가 가능한 프레임이다.

즉, I-VOP은 MPEG-4 파일에서 필수적으로 필요한 프레임이다. I-VOP을 추출하여 적용 가능한 방안은 고속 재생 기능이나, 암호화 과정을 거쳐 저작권을 보호하는데 적용될 수 있다. 고속 재생의 경우 문제가 되는 점은 대역폭이 높아지는데, 즉 2배 재생의 경우



[그림 1] MPEG 비디오 스트림의 프레임 스퀀스

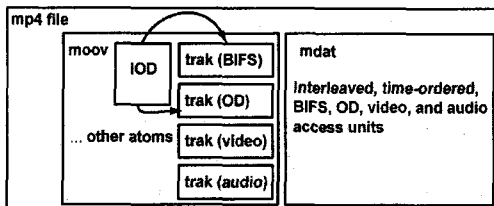
에는 정상 속도 재생시보다 두 배의 대역폭이 필요하다. 이러한 대역폭을 일정하게 해주기 위해서는 고속 재생시에 MPEG 스트림에서 I-VOP만 추출해서 전송하면 대역폭을 낮추는데 이용할 수 있다.

2.1 DRM 적용

추출한 I-VOP에 암호알고리즘을 적용시켜, DRM 솔루션에 적용할 수 있다. MP4 파일 내부에 암호화 정보를 삽입한 후 데이터 내부에 삽입된 암호화 정보는 스트리밍 서버에 의해 DRM 서버로 전송되고, DRM 서버는 인증된 사용자에게 사용자 규칙 등을 포함한 DRM 정보를 전송하게 된다. DRM 서버는 암호화와 관련된 DRM 정보에 사용자 규칙 등의 정보를 포함하여 사용자에게 전송하고, 사용자의 플레이어 내부에 삽입된 복호화 모듈을 이용하여 VOP를 실시간 복호화한다. 사용자는 암호화된 데이터를 전송받지만, 플레이어 내부의 복호화 모듈을 이용하여 원활한 서비스를 이용가능하다.

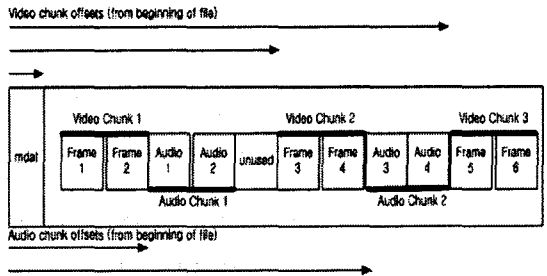
3. MPEG-4 파일 포맷의 분석

MPEG-4 System v2에서는 MPEG Movie의 시스템 파일 포맷을 규정한다. 이 시스템 파일 포맷은 MPEG-4 파일이 컴퓨터에 파일형태로 저장되는 형태를 지정하는데, 다양한 종류의 디지털 데이터를 저장하는데 편의를 주기위함이다. MPEG-4 파일은 실제 데이터를 저장하고 있는 데이터 부분과 실제 데이터의 저장 정보 및 디스크립션 정보를 담고 있는 메타데이터 부분으로 나눌 수 있다. [그림 2]은 MP4 파일 포맷 구조를 나타낸다.[4][5]



[그림 2] MP4 파일 포맷 구조

데이터 정보 가장 상위에는 Mdat Atom이 존재하고 하위로 [그림3]와 같은 실제 데이터의 집합인 Chunk가 존재한다.



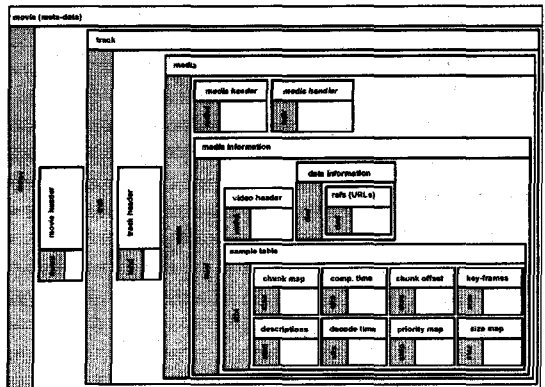
[그림 3] Chunk Offsets(실제 데이터)

메타데이터 부분에 가장 상위에는 moov atoms이 존재하는데, 이는 Tree 형식으로 하위에 여러 Atoms들을 포함한다. 이들 하위 Atoms들은 실제 데이터에 대한 세부 정보를 가지고 있는 집합이라 할 수 있다.

◆ Movie Atoms

Movie atoms 는 'moov'의 atom type을 가진다. 본 atom은 movie's data의 기술정보를 포함한다. 이 정보나 메타데이터는 atom의 다른 타입들에 저장되어진다. movie atom은 본질적으로 다른 atom을 포함한다. Movie Atoms는 가장 상위 레벨에 존재하고, 일반적으로 track atom을 포함한다.

Movie Atom은 movie의 기술정보를 정의한다. 이 정보는 응용프로그램이 본 atom에 저장된 정보를 통해 데이터를 이해하는 것을 도와준다. movie atom은 보통 movie header atom을 포함한다. [그림 4]는 Movie Atoms내부의 Atoms에 대한 세부 정보를 나타낸다.



[그림 4] Atoms에 대한 전체 세부 구조

◆ Movie header atoms

Movie header atom은 movie의 특성을 기술한다. 본 atom에 포함된 데이터는 movie의 time scale과 duration과 같은 특징을 정의한다. 본 atom의 type은 'mvhd'이다.

◆ Sample Atoms

MPEG-4 파일은 sample에 데이터를 저장한다. sample들은 미디어의 chunk들 중 하나의 chunk에 저장되는데, 이 chunk는 데이터의 임의의 접근을 위한 sample들의 묶음이다. 하나의 chunk에는 여러개의 sample들을 포함할수 있으며, chunk들의 사이즈가 다를 수 있고, chunk속의 sample들 또한 크기가 다를 수 있다. Sample을 설명하는 하나의 방법은 sample table atom을 사용하는 것이다. Sample table atom은 sample들에 관한 정보와 다른 타입의 여러 atom을 포함하는 저장소로 쓰인다. 그 여러 atom은 고유한 순서로 sample 데이터를 parse하기 위해 필요한 미디어 핸들 정보를 포함한다. 이 방식은 movie time이 중요한 actual data stream에서 data를 순차적으로 저장하지 않고도 순차적으로 실행할 수 있도록 해준다.

◆ sample Table Atoms

Sample table atom은 미디어 시간을 sample 번호나 sample 위치(location)로 변환하기 위한 정보를 포함한다. 이 atom은 또한 sample을 어떻게 해석(interpret)할지를 지시한다. 또한 track안의 미디어 sample의 시간과 인덱스를 포함한다. Table을 사용하면 시간으로 sample의 위치를 알 수 있고, sample의 type과 크기, container안에서의 offset 정보를 알 수 있다. Sample table atom을 포함하고 있는 track이 데이터를 참조하지 않으면, sample table atom은 어떤 하위 atom도 포함할 필요가 없다.

만약 track이 데이터를 참조한다면, sample table atom은 다음의 하위 atom들을 필요로 한다.

sample description, sample size, sample to chunk, chunk offset

3.1 비디오 프레임의 추출 알고리즘

I-VOP을 추출하기 위해서는 비디오 프레임(I, B, P-VOP)의 추출이 필요하다. 비디오 프레임을 추출하는 방법으로 MP4 파일의 메타데이터를 파싱하여 각 비디오 프레임의 읍셋을 추출한다.[6] MP4 파일은 실제 데이터를 가지고 있는 mdat Atom과 실제 데이

터의 기술정보를 가지고 있는 메타데이터로 나뉘어진 다. [그림5]은 문자열 파싱을 통해 MP4 파일을 Interger형으로 변환하여 각각의 Atom을 찾아 상위 4Byte를 읽어 그 Size만큼 이동하여 순차검색을 하는 루틴이다.

```

Int ConvertTypeToInt(const char* AtomType)
{
    if(strcmp("moov", AtomType) == 0)
        return moov;
    else if (strcmp("trak", AtomType) == 0)
        return TRAK;
    else if (strcmp("mdia", AtomType) == 0)
        return MDIA;
    else if (strcmp("minf", AtomType) == 0)
        return MINF;
    else if (strcmp("stbl", AtomType) == 0)
        return STBL;
    else if (strcmp("stsc", AtomType) == 0)
        return STSC;
    else if (strcmp("stco", AtomType) == 0)
        return STCO;
}
    
```

[그림 5] MP4파일에서 Atom을 찾는 루틴

1. 각 Track의 타입을 조사하여 비디오트랙인지 오디오트랙인지를 판별한다. 판별 방법은 'minf' atoms의 하위에 위치하는 'vmhd' (Video media information header atom)이 존재하는지를 확인한다. 만약 'vmhd' atoms이 존재하면 그 track이 비디오 트랙이다.
2. 비디오 트랙내에 sample table atoms의 데이터를 추출한다.
3. sample table atoms 내에 있는 sample to chunk,('stco'), sample per chunk('stsc'), sample size atom('stsz')의 각 테이블의 값을 얻어온다.

3.2 I-Frame 추출 구현

각 비디오 객체는 vop start code를 가지고 있는데, (I, B, P)Frame들은 모두 start code가 존재한다. start code는 4Byte의 값으로 (x00 00 01 B6)로 정의 되어 있다. [그림6] 비디오 데이터를 검사하는 루틴이다.

```

Int IsVopHeader(BYTE* VopHeader)
{
    if ( (VopHeader[0]&0xff) == 0x00 &&
        (VopHeader[1]&0xff) == 0x00 &&
        (VopHeader[2]&0xff) == 0x01 &&
        (VopHeader[3]&0xff) == 0xb6 &&
        (VopHeader[4]&0xc0) == 0x00 )
    
```

```

{
  return TRUE;
}
else if ( (VopHeader[0]&0xff) == 0x00 &&
          (VopHeader[1]&0xff) == 0x00 &&
          (VopHeader[2]&0xff) == 0x01 &&
          (VopHeader[3]&0xff) == 0xb3 &&
          (VopHeader[4]&0xc0) == 0x00 )
{
  return TRUE;
}
else
{ return FALSE; }
}
    
```

[그림 6] 비디오 데이터를 검사하는 루틴

또한 각 비디오 객체의 4Byte 이후 다음 2bit는 vop coding type을 정의하는데, 이 2bit를 파싱하여 (I, B, P) Frame을 구분한다.[그림 7]

```

int IsI_VopHeader(BYTE* VopHeader)
{
  if ( (VopHeader[0]&0xff) == 0x00 &&
        (VopHeader[1]&0xff) == 0x00 &&
        (VopHeader[2]&0xff) == 0x01 &&
        (VopHeader[3]&0xff) == 0xb6 &&
        (VopHeader[4]&0xc0) == 0x00 )
  {
    return TRUE;
  }
  else if ( (VopHeader[0]&0xff) == 0x00 &&
            (VopHeader[1]&0xff) == 0x00 &&
            (VopHeader[2]&0xff) == 0x01 &&
            (VopHeader[3]&0xff) == 0xb3 &&
            (VopHeader[4]&0xc0) == 0x00 )
  {
    return TRUE;
  }
  else
  {
    return FALSE;
  }
}
    
```

[그림 7] I-VOP 추출 루틴

4. 결론 및 향후연구

MPEG-4는 실시간 영상, 음향 통신이 중요 기술이 되는 영상 전화, 영상 회의 등의 분야에 가장 쉽게 적용될 수 있다. 본 논문에서는 MPEG-4 파일 포맷을 통한 비주얼 요소 중 가장 중요한 I-VOP을 추출하는 알고리즘을 설계 및 구현하여, 보다 다양한 멀티미디어 서비스에 적용하고 저작권 보호와 불법복

제방지 및 질 높은 멀티미디어 서비스의 기반을 마련해야 할 필요성이 있다.

5. 참고문헌

- [1] MPEG Korea, www.mpeg.co.kr/
- [2] The MPEG-4 video standard verification model, Thomas Sikora, Senior Member, IEEE
- [3] ISO/IEC 13818-1, Information technology- Generic coding of moving pictures and associated audio information: Systems.
- [4] The MPEG-4 Book (IMSC Press Multimedia Series
- [5] <http://www.applecomputer.co.kr/quicktime>
- [6] "QTFileFormat" Apple Computer 2000
- [7] 프로그램세계 - 멀티미디어 스트리밍