실시간 H.263 Software CODEC 구현

박대원, 이승현, 윤성규, 강태성, 이승열, 임영환
숭실대학교 컴퓨터학과

Implementaion of realtime H.263 Software CODEC

Dae-Won Park, Seung-Hyeon Lee, Sung-Kyu Yoon, Tae-Sung Kang, Seung-Yeol Lee, Young-Hwan Lim
Dept. of Computing, Soongsil University

요 약

기존의 화상전화 시스템 및 화상회의 시스템은 H/W로 구현된 H.263 CODEC을 사용한 경우는 있었지만, 컴퓨터의 성능변수와 직접적인 관계가 있는 S/W CODEC을 사용한 경우는 드물었다. 여기에 착안하여 본 연구는 PSTN 망에서의 화상회의 및 화상전화에 응용할 수 있는 실시간 H.263 S/W CODEC을 구현하였다. H.263 S/W CODEC을 MuX(Multimedia I/O Server)의 DLM으로 구현함으로써 현재 LAN 기반으로 동작하는 MuX를 PSTN망으로 확장하는데 있어 기반 기술로 사용할 수 있으며, 대역폭이 낮은 LAN 환경에서도 안정적인 화상회의가 가능하도록 하였다.

1. 서 론

기존의 화상전화 시스템 및 화상회의 시스템에서는 H/W로 구현된 H.263 CODEC을 사용한 경우는 있었지만, 컴퓨터의 성능 변수와 직접적인 관계가 있는 S/W CODEC을 사용한 경우는 드물었다.

여기에 착안하여, 본 연구에서는 PSTN 망에서의 화상회의 및 화상전화에; 응용할 수 있는 H.263 실시간 S/W CODEC DLM을 구현하였다. 실시간 S/W CODEC을 독립된 응용프로그램으로 구현한다면 개별적인 프로그램으로 사용할 수 있겠지만, 화상회의 및 화상전화 시스템에 응용하기 위해서는 H.263 스트림 제어기법 및 화상회의 Call Setup 프로그램을 함께 구현해야 한다. 위와 같은 문제점은 MuX(Multimedia I/O Server)를 사용하여 해결할 수 있다. 스트림 제어 및 화상회의 Call Setup 등 화상회의 시스템의 기반기술 모두들 MuX가 제공하므로 MuX의 DLM 형태로 S/W CODEC을 구현하였다. DLM으로 구현함으로써 실시간으로 인코딩된 H.263 및 파일 스트림과 구별 없이 디코딩이 가능하며, 화상회의 및 화상전화 시스템에 대한 다양한 응용이 가능하게 되었다.

2. 기반 기술

2.1 H.263

H.263은 저 전송률을 가지는 통신선로(64Kbps이하)에서 영상회의나 비디오 전화 등을 위한 멀티미디어 통신 서비스의 동영상 부분에 대한 압축에 쓰일 수 있는 부호화된 표현에 대한 권고 안이다. 기본적인 영상소스 코딩 알고리즘은 H.261을 기반으로 4개의 선택부호화 방법이 첨가되어 있으며, H.261에 비하여 같은 화질의 영상에 대해 거의 반정도의 비트를 생성한다. 초저속 통신망에서의 멀티미디어 서비스를 위한 터미널 제어표준인 H.324 등에서 사용되며, H.324 화상전화기는 최근 표준화된 V.34모뎀을 통해 전화망에 접속되며, 동영상의 압축은 H.261을 상당부분 개선한 H.263을 이용하고 음성의 압축은 CELP 방식인 G.723을 이용한다.

H.263이 H.261에 비해 개선된 부분을 정리하면 다음과 같다. 우선 각 매크로 블록의 움직임 벡터를 부호화 하는데 있어서 이웃하는 매크로블록의 움직임벡터와 상관도가 높음을 감안해 세 벡터의 중간 값을 취하는 보다 효율적인 방법을 사용하고 있다. 이 방법으로 약 10%의 데이터 감축이 얻어진다. 또한 한 매크로 블록 내에서 움직임을 세분화하는 블록별 움직임 추정이 가능하다.

DCT 변환계수의 효율적 양자화를 위해 양자화기를 개선해 약 3%의 데이터절약을 얻고, 또한 양자화된 변환계수들의 가변장 부호화는 H.261, JPEG, MPEG1, MPEG2의 2차원 부호를 개선해 화면내, 화면간 정보까지를 고려한 3차원부호를 사용한다. 여기서 약 5%정도의 데이터를 절약할 수 있다.

비트열의 구문(syntax)에 있어서도 기존의 H.261보다 크게 단순화해 순수한 정보 비트이외의 오버헤드를 줄이고 있다.

이밖에 성능향상을 가져오지만 복잡하여 사용여부를 옵션으로 남겨둔 기술로 구문기반 적응산술부호화와 PB프레임이 있다. 구문기반 적응산술 부호화는 복잡하기는 하지만 5~14%의 절약을 가져온다. PB프레임은 다른 기법이 비트절약을 위한 기법인데 비해 비트를 약간 더 허용하면서 초당 프레임 수를 배

2.2 MuX(Multimedia I/O Server)

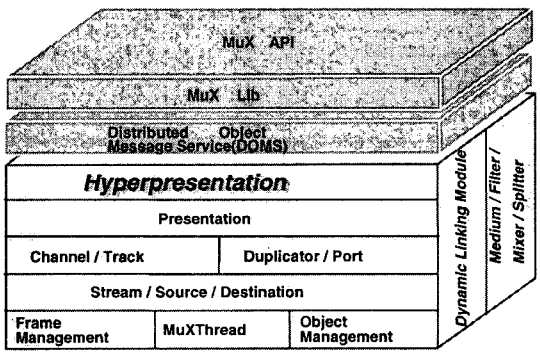
MuX는 멀티미디어 데이터 처리 및 입출력은 물론, 멀티미디어 관련 동기화 기능 등을 정의하고 있는 프레임워크를 제공한다. 또한 객체 지향 개념으로 구현된 MuX의 API를 이용하면 내부의 상세하고 복잡한 처리 사항들을 모르더라도 쉽고 빠르게 응용 서비스를 개발할 수 있다. 응용 서비스에 따른 데이터의 흐름을 살펴보면 크게 세 가지로 구분된다.

① 입력된 데이터의 내용을 저장하거나 가공하지 않고 경로만을 관장하여 출력될 곳까지 보내주는 기능

② 저장된 데이터의 통합이나 동기화를 맞추어 의미 있는 형태로 데이터의 흐름을 재구성하는 기능

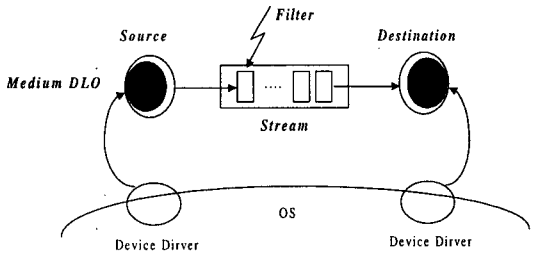
③ 사용자가 조건을 설정하여 그 조건에 만족하는 경우에 알맞은 처리를 할 수 있도록 사용자의 요구에 따른 데이터 흐름을 제어하는 기능

이러한 데이터의 처리기능은 아래 [그림 1]과 같이 MuX의 세 계층에서 제공된다.



[그림 1] MuX 서버 계층도

2.2.1 스트림 계층 (Stream Layer)



[그림 2] 스트림 계층의 데이터 흐름도

멀티미디어 정보처리 모델의 최하위 계층으로서 특정 미디어와 관련된 데이터의 흐름을 정의하고 있으며, 데이터의 근원지(source)에서 목적지(destination)까지의 전송을 담당한다.

2.2.2 프리젠테이션 계층 (Presentation Layer)

프리젠테이션 계층은 스트림 계층 위에 구성된다. 프리젠테이션은 시간 또는 공간적 동기화를 통해 구성되는 스트림들의 집합으로 볼 수 있다.

2.2.3 하이퍼 프리젠테이션 계층 (Hyperpresentation Layer)

하이퍼링크로 연결된 동적 멀티미디어 프리젠테이션 네트워크를 구성, 시간과 공간에 따른 다차원적인 멀티미디어 표현 방식을 제공하는 최상위 계층이다.

2.3 DLM (Dynamic Linking Module)

MuX는 각종 멀티미디어 디바이스를 손쉽게 지원할 수 있도록 DLO(Dynamic Linking Object) 객체로 확장할 수 있는 구조로 되어 있다. DLO 객체는 DLL(Dynamic Link Library) 기능을 사용하여 구현한다. DLL은 다음과 같은 부분으로 구성된다.

① DLL 인터페이스 함수 : MuX 프로그램에서 해당 DLL의 정보를 얻고 새로운 개체를 만들기 위해 사용되는 함수로 export된 C 함수들로 구성된다.

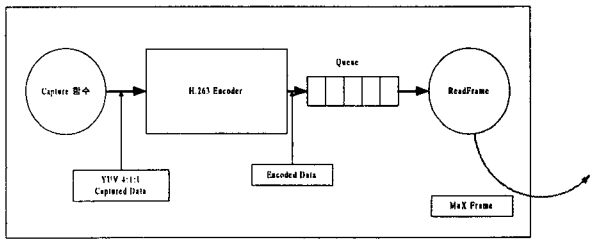
② DLO base class : 멀티미디어 입/출력 기능을 갖는 기본 객체이다. DLL 내부의 모든 객체는 DLO로 부터 상속된다. DLO는 pure 클래스이기 때문에 자체로서 객체의 개체를 가질 수 없다.

③ Medium base class : DLO로부터 상속되며 입/출력을 하나씩 갖는다.

3. 구현

3.1 H.263 인코딩 DLM

실시간으로 발생하는 캡쳐된 이미지를 사용하여 H.263 스트림을 구성하고, VFW(Video for Window) 함수를 사용하여 H/W에 대한 의존성을 제거하였다. VFW 함수를 사용함으로써 QCIF 크기의 YUV 4:1:1 planner방식을 지원하고, VFW 함수를 사용할 수 있는 캡쳐카드를 사용한다면 인코딩 DLM이 동작 가능하다.



[그림 3] H.263 인코딩 DLM 구조

H.263 인코딩 DLM 구조는 위 [그림 3]과 같으며, 각각의 부분을 살펴보면 다음과 같다.

캡쳐 함수는 콜백 함수로 구현되어 있으므로, Windows에서 자동적으로 불러 주도록 되어 있다. 캡쳐가 수행되면 이 캡쳐 함수를 통해 Windows가 캡쳐된 이미지의 포인터를 넘겨주므로 이 데이터를 복사하여 인코더에 넘겨주는 역할을 수행한다.

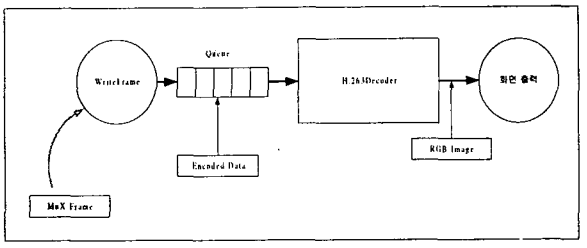
H.263 Encoder는 실제의 인코딩이 이루어지는 부분이다. 인코딩이 끝나면 인코딩된 데이터를 전역(Global) 큐(Queue)에 달아 ReadFrame에서 사용할 수 있도록 한다.

ReadFrame은 인코더에서 큐에 달아놓은 인코딩된 데이터를 하나씩 가져와 MuX 프레임(Frame)으로 만들어 MuX 스트림으로 전달하는 역할을 수행한다. ReadFrame을 거치고 나면 그

스트림을 MuX 시스템이 알아서 디코더 쪽으로 전달한다.

3.2 H.263 디코딩 DLM

H.263 디코딩 DLM은 인코더에 비해 쉽게 구현되었으며, 하드웨어와는 상관이 없는 부분이므로 인코더보다 간단한 구조를 가진다.



[그림 4] H.263 디코딩 DLM 구조

H.263 디코딩 DLM 구조는 아래 [그림 4]와 같으며, 각각의 부분을 살펴보면 다음과 같다.

WriteFrame은 MuX의 스트림을 통해 전달받은 MuX 프레임을 H.263 스트림 형태로 복원하여 디코더에 전달하는 역할을 수행한다. MuX 프레임을 H.263 스트림으로 복원한 뒤 디코더와 공통으로 사용하는 전역 큐에 달아 준다.

H.263 Decoder는 실제 H.263 스트림을 디코딩하여 YUV 데이터를 축출하는 부분이다. 인코더와는 달리 쓰레드로 구현되었다. 전역 큐에서 하나의 데이터를 빼와 이를 디코딩해 YUV 데이터를 만든다. 이 데이터를 RGB 변환을 하여 화면 출력 모듈에 전달한다.

4. 결과 및 문제점

4.1 결과

이 연구의 테스트 환경은 Cyrix 200 MMX, 96M인 시스템1과 Pentium II 350, 128M인 시스템2를 사용했으며, 캡쳐보드는 On-Air TV 카드 (BT-878 칩셋)를 사용했다.

H.263 인코딩 DLM은 시스템1에서 초당 2~3 프레임 정도의 인코딩이 가능했으며, 시스템 2에서는 약 6~8 프레임 정도의 인코딩이 가능했다.

H.263 디코딩 DLM은 인코더와 달리 두 시스템 모두 부드러운 재생이 가능했다.

4.2 H.263 인코딩 DLM의 문제점

초기 버전은 인코더를 쓰레드(Thread)로 구성하였고, 캡쳐함수와 인코더 사이에 전역 큐를 하나 더 두었다. 가장 큰 문제점은 캡쳐 함수는 초당 15 프레임을 캡쳐 함으로 엄청나게 만은 데이터가 발생한다. 이 데이터를 인코더에서 사용하는 시간이 초당 2~3 프레임 정도이므로 일정한 시간이 지나면 큐가 가득차 시스템이 다운되는 경우가 발생했다. 또한 큐에 달린 순서로 인코딩이 되므로 생각보다 큰 지연이 생겼다.

캡쳐 함수는 콜백함수이므로 Windows가 알아서 불러주는 형식이므로 쓰레드를 사용하지 않고 캡쳐 함수에서 인코더의 함수를 불러주도록 구현하였다. 이 경우 인코딩 하는 시간만큼 캡쳐 함수가 불리지 않으므로 문제가 되긴 하지만, 성능이 좋은 컴퓨터를 사용한다면 초당 6~8 프레임 정도의 인코딩이 가능하다. 또한 지연되는 시간 역시 쓰레드를 사용한 경우보다는 작다.

4.3 H.263 디코딩 DLM의 문제점

화면 출력 부분을 RGB 변환을 통해 화면에 뿌려주도록 구현하였다. 이 방법을 사용하면 어떤 그래픽 카드를 사용하는 경우든지 사용할 수 있지만, YUV 데이터를 RGB로 변환하는 시간이 걸리게 된다.

5. 결론

본 연구는 MuX DLM으로 H.263 S/W CODEC을 구현함으로써 현재 LAN 기반으로 동작하는 MuX를 PSTN망으로 확장하는데 있어 기반 기술로 사용할 수 있으며, 대역폭이 낮은 LAN 환경에서도 안정적인 화상회의가 가능하였다.

결과적으로 인코딩 및 디코딩 DLM을 구성하여 정상적인 동작을 하는 것을 확인하였지만, 처음에 예측했던 것 보다 성능이 좋지 못했다. TMN H.263은 Readability에 중점을 두고 권고안에 맞추어 구현되어 있으므로, 소스코드 상에 성능을 개선할 수 있는 여지가 충분히 보였다. 알고리즘을 개선하지 않고 Syntax만을 바꾸어도 어느 정도의 성능 향상이 가능할 것으로 보인다.

현재 S/W 인코딩은 초당 2~3프레임 (Intel Pentium II 350 : 6~8프레임)정도의 성능을 보인다. 추후 캡쳐 함수 및 CODEC 알고리즘의 개선을 성능 향상이 가능할 것이다. 또한 인코딩 코드를 Intel의 MMX Instruction을 사용하면 보다 성능이 나은 H.263 S/W CODEC을 구현할 수 있을 것이다.

6. 참고문헌

[1] 임영환, "ComBiStation : 분산 멀티미디어 컴퓨팅 환경을 위한 컴퓨터 플랫폼," 정보과학회 논문지, 제 2권, 제 1호, 1996, pp.160-181.

[2] Baker R. A. Dowring, K.Finn, E.Rennison, D.H.Kim, and Y.H. Lim, "Multimedia Processing Model for a Distributed Multimedia I/O System," Proceedings of 3rd International Workshop on Network and Operating Systems for Digital Audio/Video, 1993, pp.233-239.

[3] Rennison. E, R.Bker, D.H.Kim, and Y.H.Lim, "MuX : An X Co-Existant Time-Based Multimedia I/O Server," The X Resoure, Issue 1, 1992, pp.213-233.

[4] Image Processing Lab, University of British Columbia, TMN (H.263+) encoder/decoder, version 3.0, TMN (H.263+) codec, September 1997. See web site at http://www.ee.ubc.ca/image/.

[5] ITU Telecom. Standardization Sector of ITU, Video Codec for audiovisual services at $p \times 64$ kbits, ITU-T Recommendation H.261, March 1993.

[6] ITU Telecom. Standardization Sector of ITU, Video Coding for Low Bitrate Communication, ITU-T Recommendation H.263, March 1996.