임베디드 환경에서의 H.264/AVC 재생기 성능 분석*

권순영*O 이주경* 김영주** 정기동*
부산대학교 컴퓨터공학과* , 신라대학교 컴퓨터공학과**
{ksy2020O, jklee, kdchung }@pusan.ac.kr*, yjkim@silla.ac.kr**

Complexity Analysis of H.264/AVC Player on Embedded System

Soonyoung Kwon*O , Jookyong Lee* , Youngjoo Kim** , Kidong Chung*
Dept. of Computer Engineering, Pusan National University*
Dept. of Computer Engineering, Silla University**

요 약

동영상 압축 표준인 H.264/AVC는 압축 효율을 높이기 위해 기존의 표준과는 다른 기법들을 사용함으로 압축률은 높였지만 보다 많은 계산량을 요구한다. 제한된 자원을 가진 임베디드 환경에서는 많은 계산량은 큰 문제점이 된다. DMB를 포함한 대부분의 경우는 이를 하드웨어적으로 구현을 하고 있지만 구현비용과 업데이트의 용의를 위해서 앞으로는 소프트웨어적으로도 구현이 가능해야 할 것이다. 본 논문에서는 H.264/AVC가 임베디드 환경에서 소프트웨어로 구현을 할 경우 그에 대한 성능 평가를 수행하기 위해 실제 임베디드 장비에서 H.264/AVC 복호기와 임베디드 그래픽 라이브러리를 사용해서 재생기를 구현하였고 다양한 종류의 영상을 재생시키는 실험을 하였다. 이러한 실험을 통해 임베디드 상에서 H.264/AVC 재생기는 QCIF 화면을 초당 3프레임 정도를 재생시키는 능력을 보였다. 이는 사용자측면에서는 동영상이라고 느낄 수 없을 정도의 성능이었다. 그러므로 임베디드 환경에서 H.264로 압축된 영상을 사용할 경우에는 H.264의 프로파일이나 레벨 조정 및 프레임 넘김 기법이 필요 할 것으로 추정한다.

1. 서 론

새로운 동영상 압축 표준인 H.264는 ISO와 ITU에서 2003년 표준으로 승인되었다. 이 표준은 MPEG-2 압축 방식의 화질을 유지하면서 압축률을 50%로 낮추기 위해 가변 블록 움직임 보상, 복수 참조영상, 필터와 같은 다양한 기법을 추가하였다[1]. 동영상 스트리밍 서비스를 지원하는 응용프로그램에서 성능을 평가한 결과, H.264/AVC Main Profile의 평균 비트율은 MPEG-2에 비하여 63%, MPEG-4 Visual Advanced Simple Profile에 비하여 37% 정도 성능이 더 우수한 것으로 나타났다. 그러나 이러한 추가적인 기능으로인해 계산 복잡도는 기존의 압축 표준보다 증가하였다. H.263에 비해서 복호기의 계산 복잡도가 2배나 증가하였다.[2]

최근 국내 DMB의 표준으로 H.264가 채택이 됨으로써 임베디드 환경에서 H.264를 적용시키기 위한 연구가 활발히 진행되고 있다. 대부분의 구현이나 연구의 초점이 하드웨어적인 측면에 맞춰져 있어서 소프트웨어적으로 H.264의 복호기에 대한 성능 분석 자료가 부족하다.

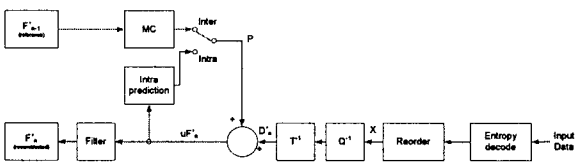
이에 본 논문은 임베디드 환경에서 H.264의 복호기를 사용하여 실질 영상을 보여주는 재생기를 구현하고 성능을 분석하였다. 분석 결과 소프트웨어적으로는 정상적인 재생기가 불가능 하였다. 이는 H.264의 계산 복잡도가 높고 임베디드 환경의 특성상 CPU의 성능이 낮기 때문이다.

본 논문의 구성은 다음과 같다. 2장에서 H.264와 임베디드 환경에서의 개발 환경 구축에 대해서 알아보고, 3장에서 재생기 구현을 설명한다. 4장에서는 재생기에서의 실험 결과를 토대로 성능을 분석하고 마지막으로 5장에서 결론을 맺는다.

2. 관련연구

2.1 H.264/AVC의 복호기

[그림 1]에서 H.264의 복호기를 블록 다이어그램으로 보여주고 있다[3].



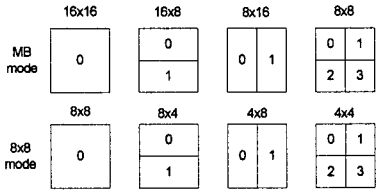
[그림 1] H.264/AVC 복호기 블록 다이어그램

기본적인 방법은 기존의 압축 표준들과 비슷하지만 몇 가지 차이점과 두 가지 과정이 추가되어 있다.

- Entropy decode : 기존 방식에서는 고정된 테이블을 사용한 반면 H.264에서는 적응적인 복호화 방식인 CAVLC 또는 CABAC 방법을 사용한다. 이러한 방식을 통해 기존의 방식보다 5~15%의 비트율을 감소시킨다.

* 본 연구보고서는 정보통신부 정보통신연구진흥원에서 지원하고 있는 정보통신기초연구지원사업의 연구결과입니다.

- T^{-1} : DCT, IDCT 과정에서 생기는 오류를 줄이기 위해서 8x8 실수형 DCT 대신 4x4 정수형 DCT를 사용한다.
- MC : 가변 블록 움직임 보상을 사용한다. [그림 2]와 같이 최소 4x4에서 최대 16x16 크기의 블록을 사용해서 움직임 예측(ME) 및 움직임 보상(MC)을 수행한다. 또한 참조 프레임은 최대 5장까지 가능하다. H.264 부호화기에서 계산량이 가장 많이 증가한 부분으로 전체 시간의 약 70%를 차지한다.



[그림 2] 움직임 보상을 위한 매크로블록 분할

추가적인 단계는 다음의 2 가지이다.

- Intra Prediction : 기존의 압축 방식에서 Intra 프레임의 경우 고정된 값의 차이만을 부호화하였지만 H.264에서 Intra 프레임의 경우 공간 중복성을 제거하기 위한 예측을 수행한다. 16x16 블록의 모드 4가지와 4x4 블록의 모드 9가지 중 가장 적합한 모드를 선택하고 움직임 예측을 수행한다.
- Filter : 블록화 현상에 의한 화질 열화를 보정하기 위한 필터가 요구된다. 블록화 현상이랑 양자화 과정에서 인접 블록간의 파라미터 및 코딩 기법의 채택 차이로 인해서 블록의 가장자리에서 경계가 나타나는 현상이다. 이를 위해 H.264에서는 인-루프 디블로킹 필터(In-Loop Deblocking Filter)를 추가하였고 주관적인 화질 향상뿐만 아니라 5~10% 비트율 감소 효과를 얻었다.

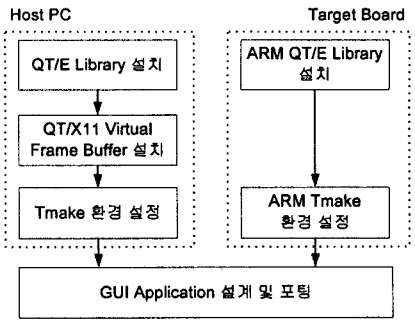
2.2 임베디드 환경에서의 개발 환경

본 논문에서 사용한 임베디드 장비는 한백전자에서 제작한 교육용 임베디드 보드이다[4].

[표 1] 임베디드 보드의 시스템 사양

모델명	HBE-EMPOS II
CPU	Intel Xscale PAX255
메모리	64Mbyte SDRAM 32 Mbyte Flash memory
주변 장치	6.4 인치 터치 스크린 LCD 패널 텍스트 LCD, 7-세그먼트, LED

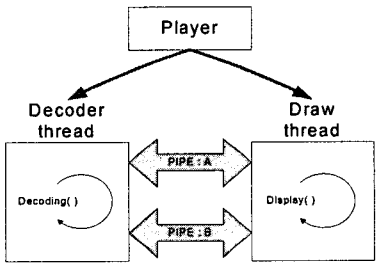
[표 1]에서와 같은 사양의 보드에 영상을 재생하기 위해 QT/E를 사용하였다. QT/E를 위한 개발 환경은 [그림 3]과 같다. 먼저 호스트 PC에서 QT/X11을 설치한 후 에뮬레이션을 위한 QT Virtual Frame Buffer를 설치하고, QT/E를 크로스 컴파일 하여 보드에 포팅 가능한 QT/E 라이브러리를 생성한다. 응용프로그램을 크로스 컴파일하면 NFS(Network File System)을 이용한 실행이 가능하다.



[그림 3] QT/E 개발 환경

3. 재생기 구현

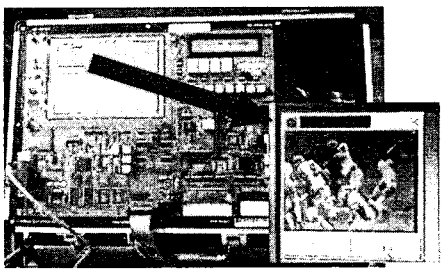
재생기는 [그림 4]과 같이 크게 두 부분으로 이루어져진 압축된 데이터를 복호화하는 부분과 복호화된 데이터를 LCD에 재생하는 부분이다.



[그림 4] H.264/AVC 재생기 모듈 구성도

[그림 4]에서 왼쪽 블록은 공개 소스인 H.264 JM98[5]을 임베디드 보드로 포팅하고 쓰레드 시작 시 압축된 데이터를 계속 복호화하는 과정이다. 오른쪽 블록은 앞에서 언급한 QT/E를 사용해서 복호화된 프레임을 화면에 재생하는 쓰레드이다. 재생하는 시간보다 복호화 하는 시간이 더 많이 걸리므로 복호기에서는 계속 복호화를 하고 재생 쓰레드에서 복호화된 데이터가 생길 때마다 화면에 보여준다. 두 쓰레드 사이에는 명령어를 전송하기 위한 PIPE:A와 복호화된 데이터를 전송하기 위한 PIPE:B가 존재하여 프로세스 통신이 이루어진다.

[그림 5]에서는 LCD 패널에 Football QCIF 영상을 보여주는 재생기를 보여준다.



[그림 5] 실제 구현된 H.264/AVC 재생기 모습

4. 성능 분석

4.1 실험 환경

[표 1]의 사양을 가진 임베디드 보드에 리눅스 커널 2.4.19을 포팅하였고 [표 2]와 같이 압축된 영상을 사용하였다.

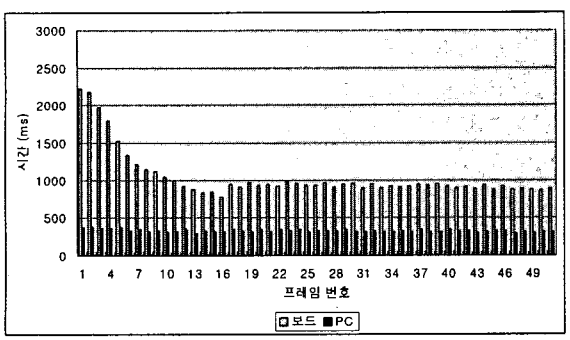
[표 2] H.264/AVC 부호화 매개변수

GOP	100(IPPP..)	프레임수	100
프레임율	30프레임/초	비트율	5,10,15 KB/S
QCIF 영상	Football Foreman Akiyo	CIF 영상	Mobile Tempete Paris

[표 2]에서는 임베디드 장비와 일반 PC 에서의 성능 평가를 보여주고 있다. QCIF 화면의 경우는 약 100ms 차이를 보이고 CIF의 경우 약 3배의 차이를 보이고 있다. 이를 통해 계산량이 많을 수 록 두 장비 사이에 시간차이가 더 증가함을 알 수 있다. [그림 6]에서는 CIF영상 Mobile에 대한 프레임별 재생 시간을 보여준다.

[표 3] PC와 임베디드 보드에서의 H.264/AVC 복호기 시간비교(ms) ; 비트율 10KB/S

	PC	보드		PC	보드
Football	220	303	Foreman	218	312
Mobile	320	937	Tempete	300	803



[그림 6] PC와 임베디드 보드에서의 프레임별 재생시간(ms)

4.2 비트율의 변화에 따른 성능 분석

[표 3]에서는 비트율 변화에 따른 재생 시간을 보여준다. QCIF 영상의 경우는 약 300ms의 재생시간이 필요하고 CIF의 경우 약 800ms 정도의 시간이 걸리는 것을 볼 수 있다. 한 프레임의 재생 시간이 300ms이면 초당 3프레임정도를 보여 줄 수 밖에 없는 성능이다. 이는 일반사용자들이 동영상이라고 느낄 수 없을 정도의 성능이다. 또한 CIF의 경우는 초당 1프레임을 보여 줄 수 있고 또한 현재 대부분의 핸드폰이나 소형의 임베디드 장비는 실험용 보드보다 성능이 더 낮음을 고려할 때 이러한 성능으로는 H.264 기반의 동영상 재생이 힘들 것이다.

[표 3] 비트율 변화에 따른 재생 시간 ;(a):비트율이 5(KB/S)일 경우, (b):비트율이 10(KB/S)일 경우, (c):비트율이 15(KB/S)일 경우에 대해서 100프레임을 재생하는데 걸리는 평균시간(ms)

		(a)	(b)	(c)
QCIF	Foreman	238	324	362
	Football	252	319	377
	Akiyo	236	271	290
CIF	Mobile	944	944	945
	Tempete	804	811	831
	Paris	671	757	792

4. 결론

본 논문에서는 임베디드 환경에서 H.264/AVC를 위한 재생기를 구현, 성능 분석을 수행하였다. 실험을 통해 재생기의 성능을 평가한 결과 아직 소프트웨어적으로 H.264 기반의 동영상을 재생하기에는 많은 어려움이 따른다. 현재의 임베디드 환경에서 소프트웨어적으로 구현을 할 경우 낮은 레벨과 프로파일을 쓰는 것만으로는 해결이 H.264 기반의 동영상 재생은 힘들다. 추가적인 코덱의 수정 또는 멀티미디어 전송 기법 중 프레임 넘김 기법이 추가적으로 필요할 것이다. H.264는 기존의 방법과는 다르게 4x4 정수형 DCT를 사용하고 블록 크기가 유동적이므로 현재 제안된 프레임 넘김 기법은 적용이 불가능하다. 향후 프레임 넘김 기법에 대한 연구가 필요 할 것이다.

[1] T.Wiegand, G.J.Sullivan, G. Bjontegaard, and A.Luthra, "Overview of the H.264/AVC video coding standard,"IEEE Trans. Circuits Syst. Video Technol.,vol. 13, pp. 560-576, July 2003
[2] V. Lappalainen, A. Hallapuro, and T.D. Hamalainen, "Complexity of Optimized H.26L Video Decoder Implementation," IEEE CSVT, vol 13., pp 717-725, July 2003
[3] Iain E.G. Richardson, "H.264 and MPEG-4 Video Compression" WILEY , 2003
[4] http://www.hanback.co.kr/htm/sub2_2.htm
[5] http://iphome.hhi.de/suehring/tml/download/