Journal of Korea Multimedia Society Vol. 9, No. 1, January 2006(pp. 59-72)

예측주기를 이용한 동적 프레임 삭제 트랜스코딩 기법

김성민[†], 김영주[††], 박성호[†††]

요 약

유·무선 네트워크의 기술의 발전과 단말기의 소형화 기술은 이동 환경에서 다양한 단말기를 이용하여 멀티미디어 콘텐츠를 이용할 수 있게 만들었다. 그러나 네트워크의 구성과 단말기들의 성능이 다양해짐에 따라, 동일한 품질의 컨텐츠를 다양한 이동 환경에 위치한 사용자에게 서비스를 제공하는 것은 매우 어려운 일이다. 그러므로 멀티미디어 스트리밍 시스템은 사용자의 네트워크와 단말기 환경을 고려하여 적응적으로 서비스를 제공하여야 한다. 트랜스코딩 기법은 다양한 네트워크 환경에서 적응적인 비디오 스트리밍 서비스를 제공하기 위한 하나의 방법이다. 본 논문은 다양한 네트워크 환경에서 비디오 스트림을 서비스하기 위해 예측주기를 이용한 동적 프레임 삭제 트랜스코딩 기법을 제안한다. 제안된 트랜스코더에서는 부호기의 예측주기와 프레임율을 조절하는 삭제주기를 이용하여 트랜스코딩 과정에서 소요되는 시간을 최적화 할 수 있다. 실험 결과, 기존의 동적 프레임 삭제 트랜스코딩 기법에 비해서 본 논문에서 제안한 기법은 PSNR값은 유사한 반면에 계산 복잡도에서 매우 높은 성능 향상을 보였다.

A Dynamic Frame-skipping Scheme for Transcoder Using the Prediction Period

Sungmin Kim[†], Youngju Kim[††], Seongho Park[†††]

ABSTRACT

The rapid growth of technology for network and terminal devices is able to service multimedia contents in mobile environments. But as the environment of networks and the performance of devices is varied, the same quality service is very difficult to be provided for user in internet and mobile environment. Therefore, multimedia streaming system must provide an adaptive service in considering network and device environment. Video transcoding techniques are the good solution that can provide video streaming service in various network environments adaptively. This paper proposes a dynamic frame-skipping transcoding scheme to provide adaptive streaming service in various network environments. This scheme uses the prediction period for transcoding. A proposed transcoding scheme can reduce time complexity through combination of prediction period and skipping period to control frame-skipping rate in encoder. In simulation results, the performance of proposed scheme is similar to the performance of a traditional dynamic frame-skipping scheme in PSNR value. But the performance of proposed scheme outperforms the performance of traditional scheme in time complexity.

Key words: Transcoding(트랜스코딩), Streaming System(스트리밍 시스템), Prediction Period(예측주기), Time Complexitv(시간복잡도)

※ 교신저자(Corresponding Author) : 박성호, 주소 : 부산광역시 금정구 장전동 산30번지(609-735), 전화 : 051)510-2785, FAX : 051)581-4692, E-mail : shpark@pusan.ac.kr
접수일 : 2005년 5월 9일, 완료일 : 2005년 9월 1일
[†] 부산대학교 컴퓨터공학과 박사과정
(E-mail : morethannow@pusan.ac.kr)
[††] 신라대학교 컴퓨터공학과 조교수
(E-mail : yjkim@silla.ac.kr)
[†††] 종신회원, 부산대학교 정보전산원 조교수
※ 본 연구는 부산대학교 교내학술연구비(4년 과제)에 의한 연구임.

1. 서 론

유·무선 네트워크의 기술과 단말기의 소형화 기술이 발전함에 따라, 이동 환경에서 다양한 멀티미디어 응용 서비스를 이용할 수 있게 되었다. 오늘날의 멀티미디어 응용 기술은 다양한 대역폭을 가진 네트워크를 통해 홈쇼핑, 게임, 주문형 비디오와 같은 다양한 서비스를 제공할 수 있다[1,2]. 이러한 응용에서 비디오 스트리밍 기술은 미디어 전송에서 중요한 역할을 담당하고 있으며, 연구 기관, 대학 및 기업체 등에서 표준과 새로운 기술에 대한 연구 개발을 수행하고 있다.

네트워크와 단말장치가 다양해짐에 따라 동일한 컨텐츠 제공을 위해 다양한 포맷이 요구된다. 트랜스코딩은 컨텐츠를 하나의 스트림으로 다양한 네트워크에 연결된 이질적인 단말장치에 서비스를 제공한다. (그림 1)은 비디오 트랜스코더를 이용한 스트리밍 서비스를 나타낸다. 비디오 트랜스코딩은 임의의 포맷, 크기, 및 비트율로 부호화된 입력 비디오 스트림을 이질적인 네트워크 환경의 클라이언트에게 효과적으로 서비스하기 위하여 다른 속성의 비디오 스트림으로 변환하는 기술이다[9-11]. 부호화된 입력 스트림의 속성을 변경하기 위해서는 복호화와 재부호화 과정이 동시에 필요하므로, 비디오 트랜스코더는 (그림 2)와 같이 복호기와 부호기로 구성된다.

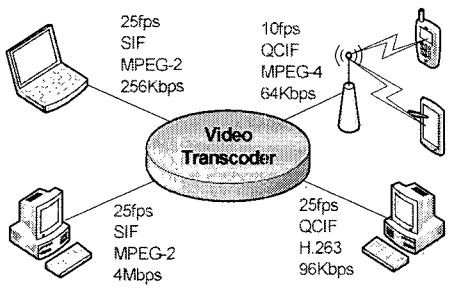


그림 1. 비디오 트랜스코더를 이용한 스트리밍 서비스

비디오 트랜스코딩은 입, 출력 스트림의 부호화 규격을 기준으로 크게 동기종 변환과 이기종 변환의 두 범주로 나눌 수 있다[12]. 동기종 비디오 트랜스코딩은 출력 스트림의 부호화 규격을 입력 스트림과 동일하게 유지하며 양자화 인자, 프레임율, 해상도를 변환하는 기법이다[8]. 반면에, 이기종 비디오 트랜스 코딩은 출력 스트림의 부호화 규격 및 속성을 입력과 다른 포멧으로 변환하는 기법이다.

동기종 비디오 트랜스코딩에서 양자화 인자 변환 방법은 복호화 후의 부호화 과정에서 적절한 양자화 인자를 선택하는 간단한 방법으로 비트율을 조절한다[13,14]. 그러나 트랜스코더 내부 부호기의 재양자화로 인해 화질 열화가 발생하는 단점이 있다[15,16]. 프레임율 조절 방법은 입력 스트림의 프레임을 반복적으로 제거하여 비트율을 조절한다. 이 기법은 가용 대역폭이 불충분한 환경의 클라이언트를 지원하거나 전송되는 프레임들의 공간적 화질을 높일 수 있다[8]. 마지막으로 해상도를 변환하는 기법은 이동전화기, PDA 및 Lap-top과 같이 디스플레이 화면의 크기에 제약을 가진 단말기를 사용하는 클라이언트를 지원하기 위해 입력된 스트림의 화면 크기를 조절한다[3].

프레임율을 조절하는 비디오 트랜스코더는 클라이언트가 속한 열악한 네트워크 환경 또는 프레임의 공간적 화질 향상을 위해서 입력된 비디오 스트림의 프레임을 반복적으로 제거하여 비트율을 조절한다. 그러나 비트율을 줄이기 위해 단순하게 프레임을 제거하면 프레임 간의 시간적 연관성을 가진 프레임이 손실되어 화질 열화나 에러 축적 현상이 발생할 수 있다. 그러므로 움직임 예측 과정을 통하여 손실된 시간적 연관성을 보상해야 한다[18,19].

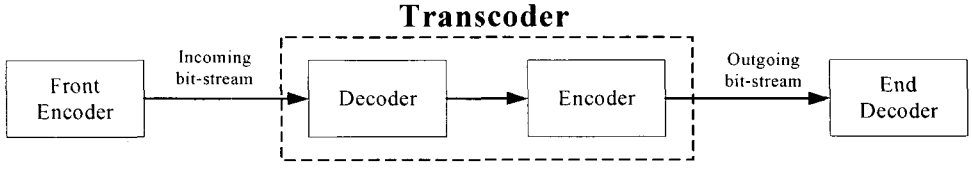


그림 2. 비디오 트랜스코더의 구조

움직임 예측은 프레임율 조절 트랜스코더에서 제거된 프레임의 움직임 벡터를 사용하지 않고 현재 프레임과 제거되지 않은 가장 가까운 순방향 프레임 간의 새로운 움직임 벡터를 다시 예측하여 프레임 간의 시간적 연관성을 얻는 방법이다. 그러나 재 예측 기법[4]은 움직임 벡터를 재 계산함으로 정확성에 비해 많은 계산 복잡도를 요구한다. 움직임 벡터 재사용 기법은 움직임 벡터를 다시 계산하지 않고 제거된 프레임의 움직임 벡터를 재사용하여 프레임 간의 시간적 연관성을 추출한다[5,18,19].

이질적인 네트워크 환경에서 다양한 클라이언트를 지원하기 위한 대부분의 트랜스코딩 기법은 입력된 비디오 스트림의 화질 열화을 인간의 시각 시스템이 허용하는 범위 내로 쉽게 조절할 수 있지만, 계산 복잡도의 최소화에는 한계가 있다. 본 논문에서는 이러한 단점을 극복할 수 있는 부호기와 이에 따른 복호기로 구성되는 스트리밍 시스템을 위한 트랜스코딩 기법을 제안한다. 이 시스템 안에서의 트랜스코더는 계산 복잡도를 최소화시켜 프레임율을 조절할 수 있다. 실험을 통한 결과에서 제안된 스트리밍 시스템은 기존의 방법에 비해 비슷한 출력 화질과 약 8%~65%정도의 계산 복잡도로 향상된 성능을 보였다.

본 논문의 구성은 다음과 같다. 2장에서는 기존의 트랜스코더 구조와 트랜스코딩 알고리즘에 관한 연구들에 대해 살펴본다. 3장에서는 제안한 서버 부호기의 알고리즘과 움직임 벡터 합성 알고리즘을 설명하고, 4장에서는 제안한 비디오 스트리밍 시스템의 성능을 평가하고 분석한다. 마지막 5장은 본 논문의 연구 내용을 요약하고 향후 연구 방향을 제시한다.

2. 관련 연구

트랜스코딩에 관한 대부분의 연구는 트랜스코더의 구조 또는 움직임 예측 알고리즘을 중심으로 수행되었다[8]. 이는 트랜스코더 내부 처리 과정에 따라 계산 복잡도와 비디오 화질의 편차가 다를 수 있고, 개선된 움직임 재사용을 통해 화질 및 계산 복잡도에서 성능을 향상시킬 수 있기 때문이다.

2.1 트랜스코더 구조

2.1.1 CPDT(Cascaded Pixel-Domain Transcoder)

효율적인 트랜스코드는 비트율을 감소시키고 가능한 계산 복잡도를 낮게 유지하면서 높은 화질을 얻는 수 있어야 한다. 즉, 비트율이 감소된 스트림은 비트율만 감소되고 화질 저하는 없어야 이상적이라 할 수 있다. 이것을 위한 가장 일반적이고 직접적인 방법은 (그림 3)에 나타나 있는 것처럼 입력된 비디오 스트림을 완전히 복호화하여 새로운 비트율로 재부호화하는 것이다[8]. 그러나 이 기법은 높은 화질을 보장하는 반면 높은 계산 복잡도를 요구하므로 실시간 비디오 스트리밍 서비스에 적용될 수 없는 단점을 가진다.

2.1.2 DDT(DCT-Domain Transcoder)

비트율 감소를 위해서 부호화와 복호화 과정은 반드시 필요하지만 너무 높은 계산 복잡도는 실시간 스트리밍 서비스에 장애가 된다. 비디오 트랜스코딩의 움직임 벡터 예측 과정은 많은 시간이 소모되므로 계산 복잡도를 줄이기 위하여 제안된 구조가 (그림

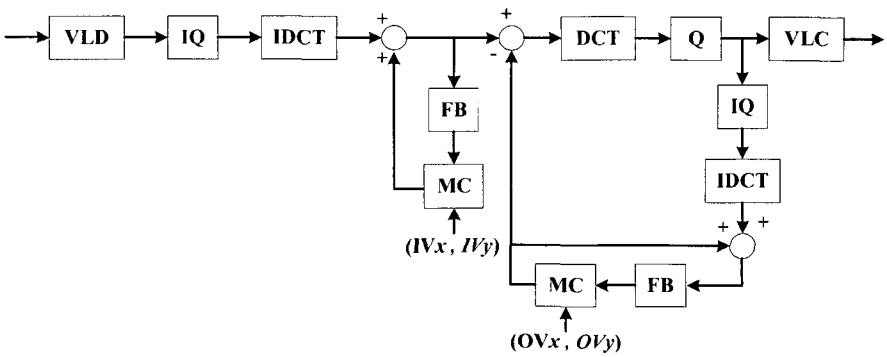


그림 3. 픽셀 영역 기반의 트랜스코더

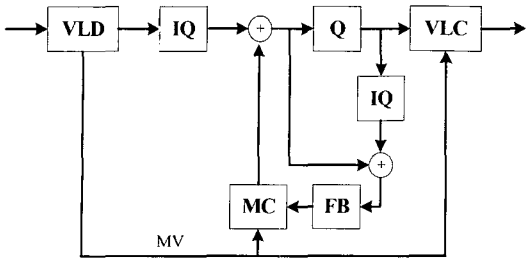


그림 4. DCT 영역 기반의 트랜스코더

4)에서 설명된 DCT-Domain Transcoder 이다. 이 트랜스코더는 CPDT에 비해 간단한 구조이고 낮은 계산 복잡도를 가지므로 많은 연구에 사용되고 있다. 그러나 움직임 예측 과정의 부재는 복호화된 스트림과 부분적으로 재 생성되어 부호화된 스트림 간의 불일치를 초래하여 에러 축적 현상을 발생시키며, 이로 인해 화질 열화 현상이 발생한다.

2.1.3 Dynamic Frame Skipping Transcoder

픽셀 영역 기반의 트랜스코딩에서는 높은 계산 복잡도를 극복하기 위하여 간소화된 구조가 많이 연구되었지만 줄어든 계산 복잡도로 인해 프레임 간의 시간적 연관성의 부정확성으로 인한 화질 열화 현상의 발생은 필수적이다[18]. 양자화 인수의 증감을 통하여 원하는 비트율로의 조절이 가능한 구조는 에러 축적으로 인하여 연이은 프레임의 화질 열화가 발생하므로 높은 화질을 유지할 수 없다. 그러므로 최근에는 양자화 에러를 인간의 시각 시스템이 허용할 수 있는 범위 내로 용인하면서 계산 복잡도를 줄이는 구조에 대한 연구가 많이 이루어지고 있다[5].

양자화 에러나 높은 계산 복잡도를 개선하기 위하여 매크로블록의 모드에 기반한 트랜스코더 기법[17]이 제안되었다. 이 기법은 스위치로 매크로블록의 모드를 식별하고 해당 모드가 MC 또는 Non-MC 인지에 따라 트랜스코딩 기법을 달리 적용할 수 있어서 적은 계산 복잡도와 높은 화질을 얻을 수 있을 뿐만 아니라 클라이언트의 가용 대역폭에 따라 비디오 스트림의 프레임율을 동적으로 조절할 수 있는 구조이다. MC 모드로 부호화된 매크로블록은 움직임 벡터가 지정하는 영역이 이전 프레임 내에 있는 임의의 여러 매크로블록들에 걸쳐져 있지만 Non-MC 모드의 매크로블록은 움직임 벡터가 이전 프레임 내에 있는 하나의 임의의 매크로블록과 정합된다.

(그림 5)는 매크로블록의 모드에 기반하여 동적인 프레임율 조절이 가능한 트랜스코더의 구조를 보여준다. 비디오 스트림이 입력되면 먼저 VLD(Variable Length Decoder)는 매크로블록의 헤더 정보, 부호화 모드, 움직임 벡터, 그리고 양자화된 DCT 계수를 추출한다. 양자화된 DCT 계수는 역양자화기(Inverse Quantizer)를 거치면서 DCT 계수로, 역DCT (Inverse Discrete Cosine Transform)를 통해 픽셀 값으로 바

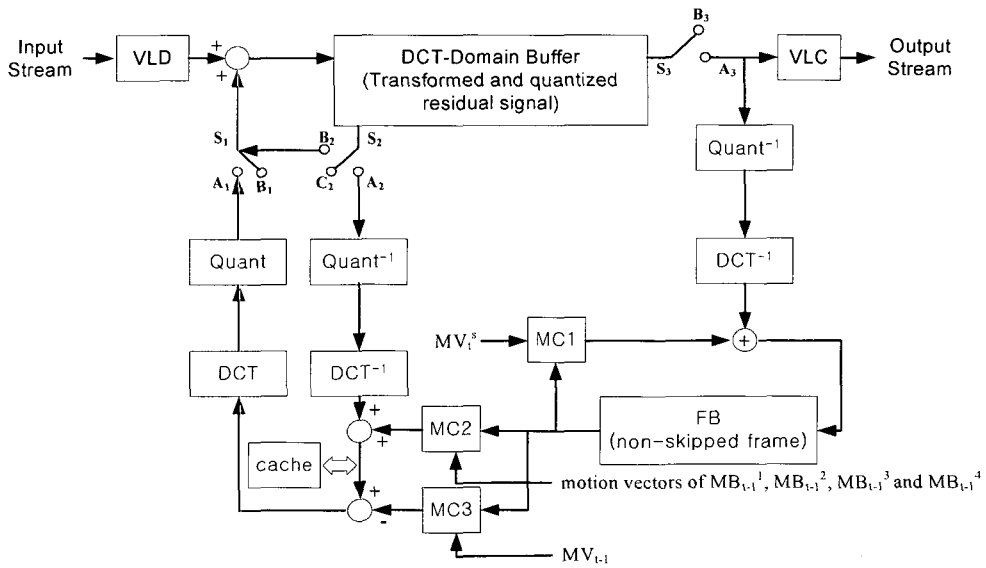


그림 5. 매크로블록 모드에 적응적인 트랜스코더

뀌게 된다. 복호화의 최종적인 값인 현재 프레임의 픽셀 값은 이전 프레임과의 차이 값으로 (그림 5)의 DCT-Domain Buffer에 저장된다. 일반적으로 비디오 스트림의 트랜스코딩에서 출력 비트율은 대개 입력 비트율보다 낮다. (그림 5)의 동적 프레임율 조절을 위해 SW(Switch)가 사용되는데, 현재 매크로블록의 차이 값이 저장되는 DCT-Domain Buffer를 업데이트하기 위해 각 매크로블록의 부호화 모드 구분에 (표 1)과 같이 SW1과 SW2가 적용되고 프레임 제거에는 (표 2)와 같이 SW3이 사용된다.

(그림 5)의 비디오 트랜스코더는 Non-MC로 부호화된 매크로블록을 픽셀 영역의 값으로 복호화하지 않고 DCT-Domain Buffer에 저장된 값에 가산만으로 처리하여 전체적인 계산 복잡도를 낮출 수 있다. 이 방법은 하나의 비디오 스트림에서 Non-MC 매크로블록의 비율이 높은 경우에만 좋은 성능을 낼 수 있다. 그러나 대부분의 비디오 스트림에서 MC 매크로블록의 개수는 움직임의 특성에 의존적이므로 트랜스코딩의 성능은 MC 매크로블록의 처리 방법에 좌우된다. (그림 5)의 트랜스코더는 MC 매크로블록을 픽셀 영역의 값으로 추출한 후 움직임 예측 과정을 수행하는데, 정교하지 않은 움직임 예측으로 움직임이 많은 비디오 스트림에는 화질의 열화가 많이 발생하는 단점을 가진다.

2.2 움직임 재사용 알고리즘

프레임율을 조절하는 비디오 트랜스코딩에서는 움직임 예측 과정이 필요하지만, 이는 가장 높은 계산 복잡도를 차지하므로, 움직임 예측 과정의 개선이 필요하다. 그래서 최근에 진행되고 있는 많은 연구들의 방향은 움직임 벡터를 추출하는 움직임 예측 과정

표 1. 프레임율 조절 모드에 따른 SW3의 위치

프레임 조절 방법	SW3 위치
삭제되는 프레임	B3
남겨진 프레임	A3

표 2. 부호화 모드에 따른 스위치 SW1과 SW2의 위치

부호화 모드	SW1 위치	SW2 위치
MC	B1	B2
Non-MC	A1	A2

을 경량화하면서 비디오 화질의 열화를 최소화하는 것이다. 움직임 예측의 목적은 (그림 6)에 묘사되어 있는 블록 정합 알고리즘(Block Matching Algorithm)을 사용하여 현재 매크로블록이 참조하는 프레임의 매크로블록의 위치를 일정한 범위(±1~16 pixels)에서 찾는 것이다. 가장 적합한 블록을 찾기 위해서 MAE(Mean Absolute Error)가 최소인 블록을 찾고 블록 위치의 차이값을 움직임 벡터로 사용한다.

프레임율을 조절하는 트랜스코더에서 새롭게 구성된 비디오 스트림 프레임 간의 시간적 연관성을 위해 움직임 벡터 예측을 다시 수행하는 방법을 재예측이라 한다[4]. 이를 위한 기법은 LOG, PHODS, 및 EPZS 등이 있다. 그러나 이들 기법들은 매우 높은 계산 복잡도를 요구하므로 본 논문에서는 고려하지 않았다.

현재의 매크로블록이 이전 프레임의 매크로블록을 참조하고 있을 때, 현재 블록은 이전 블록의 한 개, 두 개 또는 네 개에 중첩되어 있을 수 있다. 각각의 매크로블록은 하나의 움직임 벡터를 가지므로, 한 개의 매크로블록에 정합되는 경우를 제외하고는 중

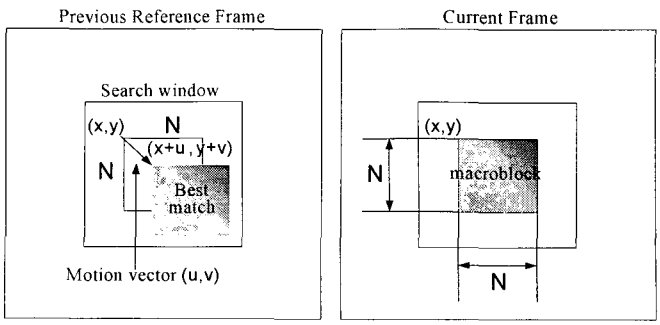


그림 6. 블록 정합 알고리즘(Block Matching Algorithm)

첩된 여러 개의 매크로블록 중에서 하나를 선택해야 한다[18,19]. (그림 7)과 같은 입력 스트림 구조에서 단순히 프레임 하나만 삭제되는 경우의 MV'는 현재 프레임의 MV1과 삭제되는 프레임의 MV2를 단순히 더하여 재사용할 수 있다. 이 기법은 움직임 재 예측을 하지 않으므로 계산 복잡도를 줄일 수 있지만 현재 매크로블록이 네 개의 인접 매크로블록 영역을 일부분씩 참조하고 있으므로, 하나의 매크로블록을 선택하는 기준이 중요한 이슈가 된다.

2.2.1 Bilinear Interpolation

프레임율을 조절하는 트랜스코딩에서 Bilinear Interpolation[5]은 (그림 8)에 나타난 것처럼 제거되는 프레임의 모든 움직임 벡터를 저장하고 역방향 순서로 움직임 벡터를 보간한다. 이것은 AMVR (Adaptive Motion Vector Resampling) 기법에서 제시된 수식을 사용하여 각각의 매크로블록의 0이 아닌 AC 계수의 개수로 공간적 활동량을 합성하여 새로운 움직임 벡터를 도출한다[3]. DC와 AC 계수는 이전 영상과의 차이 값이 DCT 변환된 주파수 영역의 값이므로, 63개의 AC 계수 중에서 0이 아닌 값이 많다는 것은 이전 영상과의 차이가 크다는 것을 의미한다. 그러므로 0이 아닌 AC 계수의 개수는 DCT 에너지 또는 공간적 활동량의 의미로 종종 언급된다. 그러나 이 기법은 제거되는 프레임의 모든 움직임 벡터를 저장해야 하는 메모리 문제, 합성된 움직임 벡터가 중첩되는 네 개의 매크로블록이 차지하는 영역의 크기를 고려하지 않아 정확성이 떨어지는 단점을 가지고 있다.

2.2.2 FDVS(Forward Dominant Vector Selection)

FDVS 기법에서는 현재의 매크로블록이 참조하는 프레임의 여러 매크로블록에 중첩되어 있을 때, 가장 넓은 공간적 영역을 차지하는 매크로블록의 움직임 벡터를 선택한다[19]. 예를 들어, (그림 8)에서 Frame(n-1)이 제거되었을 경우 MB1'이 차지하는 공간에서 오른쪽 상단에 위치한 매크로블록의 움직임 벡터가 선택된다. 프레임이 두 개 이상 제거될 경우에도 적용될 수 있는데, Frame(n-1)과 Frame (n-2)가 제거되었을 경우에 MB1의 움직임 벡터는 $I_1(n)+I_2(n-1)+I_3(n-2)$로 구성할 수 있다.

2.2.3 ADVS(Activity Dominant Vector Selection)

FDVS 기법은 움직임 벡터를 재사용하여 계산 복잡도를 줄일 수 있는 방법으로 하나 또는 연속적으로 여러 프레임들이 제거될 경우에 쉽게 사용될 수 있지만, 움직임 예측 과정을 거치지 않고 움직임 벡터를 선택한 후에 단순히 재사용하므로 프레임 간의 시간

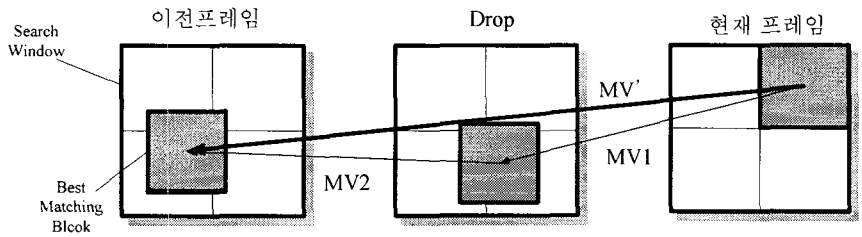


그림 7. 움직임 벡터의 재구성

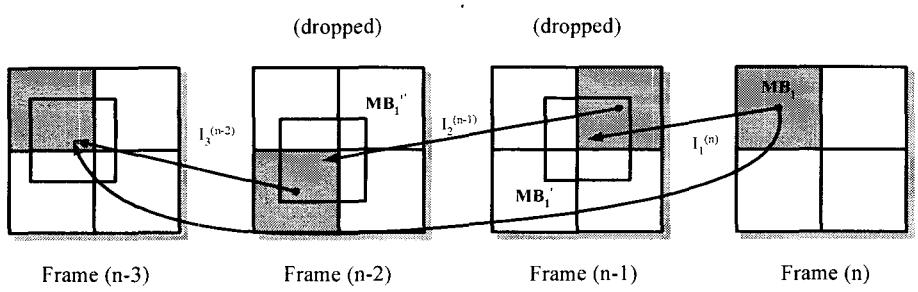


그림 8. FDVS 기법의 움직임 벡터 재구성

적 연관성을 이용할 수 없는 단점을 가진다. 그리고 현재의 매크로블록이 두 개 이상의 매크로블록에 중첩되어 있고 그 영역의 크기가 모두 같을 경우에는 기준에 따라 움직임 벡터를 선택하기 어렵다. 앞서 설명된 블록 정합 알고리즘은 일반적으로 객체의 경계에서 더 큰 오차를 발생시키는 경향이 있다. 따라서 중첩되는 매크로블록 영역만으로 움직임 벡터를 단순하게 선택하는 FDVS 기법은 예측 오차를 발생시킬 가능성이 높으므로 움직임 예측의 정확도가 떨어진다. ADVS 기법에서 DCT의 에너지는 8×8 블록 기준으로 0이 아닌 DCT 계수의 개수를 이용하여 측정되므로 DCT 에너지는 하나의 블록에서 0이 아닌 DCT 계수의 개수에 비례한다[18]. 즉, ADVS 기법에서 현재 매크로블록과 중첩된 이전의 매크로블록 중에서 유력한 것을 선택하는 기준은 각 매크로블록의 DCT 에너지(또는 활동상태 정보량)이다. 0이 아닌 DCT 계수의 개수는 별도의 변환없이 입력된 비디오 스트림에서 추출할 수 있으므로 추가적인 계산을 필요로 하지 않는다.

3. 예측주기를 이용한 동적 프레임 삭제 트랜스코딩 기법

3.1 예측주기를 이용한 부호기

일반적으로 프레임율을 조절하는 트랜스코더에서 가장 큰 이슈는 시간적인 연관성이 있는 프레임간의 예측 관계를 재구성하는 것이다. 그러므로 대부분의 트랜스코더에서 삭제되는 프레임의 움직임 벡터를 재사용하여 움직임 예측에 활용한다. 그러나 기존의 알고리즘은 트랜스코더 내부에만 한정되어 있으므로 계산 복잡도를 줄이는 데 한계가 있다.

본 논문에서는 기존의 서버 부호기에 "예측주기"를 이용하여 트랜스코더에서 제거될 프레임의 부가적인 처리를 줄일 수 있는 방법을 제안한다. 예측주기는 참조되는 프레임이 나타나는 프레임의 간격을 말한다. 예측주기를 스트림에 적용하면 연이은 프레임에 참조되지 않는 프레임이 있으므로 프레임율 조절과정에서 삭제되는 프레임에 소모되는 계산 복잡도를 줄일 수 있다. 예를 들어, 예측주기의 시작 프레임은 예측 구간에서 "예측주기-1"개의 프레임이 모두 참조하고 있으므로, 구간 내의 하나 혹은 여러 개의 프레임이 삭제되더라도 참조되는 프레임이 부호기 내 버퍼에 저장되어 반복적으로 재사용될 수 있다.

(그림 9)는 비디오 트랜스코더에 입력 스트림을 제공하는 서버의 부호기를 나타내고 있다. 예측주기에 따라 구성되는 비디오 스트림의 구조는 변경된다. 연이은 프레임에 참조된다면 참조 프레임으로써 스위치는 A의 위치에, 그렇지 않다면 B의 위치에 있게 된다. 예측주기는 동적인 프레임율 조절을 위해서 여러 가지 값을 가질 수 있는데, (그림 10)에는 예측주기가 2일 경우에 생성되는 프레임의 구조가 나타나 있다. 일반적으로 참조되는 프레임의 간격이 길어질수록 참조하는 프레임의 예측 오차는 커지기 때문에 이 점을 고려하여 프레임간의 예측주기를 2, 3, 그리고 4인 경우로 제한한다.

3.2 동적으로 프레임율을 조절하는 트랜스코더

[17]에서 제안한 기법은 앞서 소개된 동적으로 프

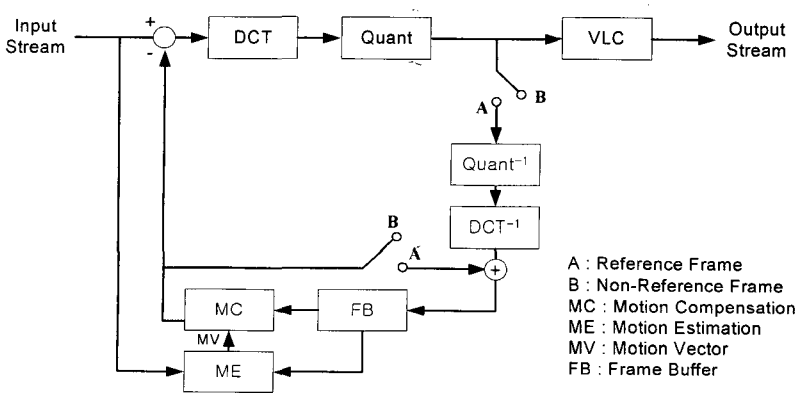


그림 9. 서버의 부호기

레임율 조절이 가능한 트랜스코더이고 그 성능은 입증된 바 있다. 각 매크로블록이 부호화된 모드, 즉 Non-MC 또는 MC에 따라 그 처리 과정을 달리하여 효율적인 트랜스코딩을 수행한다. 그러나 움직임 예측 방법으로 FDVS 기법을 그대로 사용하여 전체적인 계산 복잡도를 줄이려고 했으나 부정확한 움직임 벡터로 인해 화질의 열화를 발생시킨다.

이를 개선하기 위해 본 논문에서는 움직임 벡터 합성 기법을 앞서 설명된 동적인 프레임율 조절이 가능한 트랜스코더[17]에 적용하고 MMAT(Motion vector Mode Adaptive Transcoder)로 명명한 후 (그림 11)의 제안된 비디오 트랜스코더에 포함한다.

(그림 12, 13)는 제안된 트랜스코더에서 동적 프레임율 조절의 세 가지 예가 설명한다. 서버의 부호기에서 예측주기를 통해 비디오 시퀀스의 구조를 변경했다면, 트랜스코더에는 "삭제주기" 가 적용되어 프레임율을 조절하게 된다. 삭제주기가 적용된 프레임은 NP 또는 P 그리고 NS 또는 S의 속성이 부여되는데, NP는 처리과정이 필요없는 프레임으로 Non-Processing, P는 처리과정을 필요로 하는 프레임으

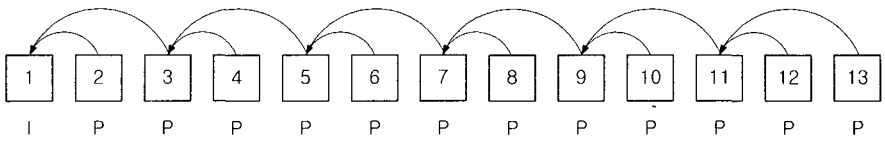


그림 10. 예측주기 2가 적용된 비디오 스트림 구조

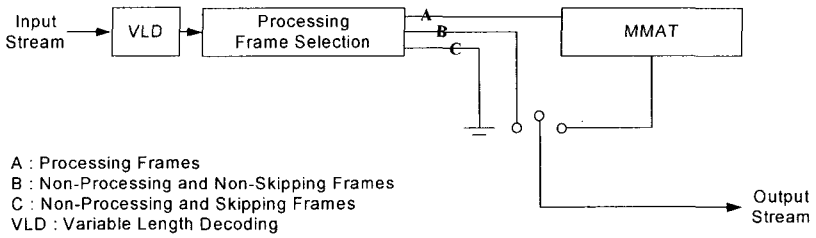


그림 11. 트랜스코더 구조

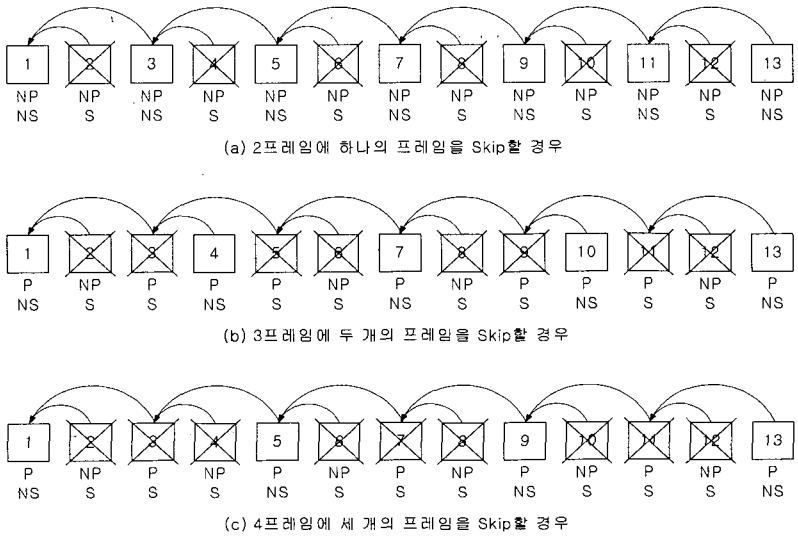


그림 12. 예측주기가 2인 프레임을 제거할 경우

로 Processing, NS는 남겨지는 프레임으로 Non-Skipping, 그리고 S는 삭제되는 프레임으로 Skipping을 의미한다.

(그림 12) (a)에서 삭제되는 프레임이 출력될 프레임에 참조되지 않으므로 부가적인 처리 과정이 전혀 필요하지 않다. (그림 13) (b)에서 2번 프레임은 삭제되어도 이후의 어떤 프레임에도 참조되지 않아서 부가적인 처리를 필요로 하지 않지만, 3번 프레임은 출력될 4번 프레임에 의해 참조되고 있으므로 삭제되는 프레임이지만 처리해야 할 프레임이 된다. 그리고 1번 프레임은 4번 프레임이 3번 프레임을 통해서 움직임 정보 및 부호화 정보를 변환할 때 참조되므로 처리해야 할 프레임이다. 기존의 트랜스코더에서는 모든 프레임이 처리 대상이지만 예측주기가 적용된 비디오 스트림을 이용하면 트랜스코딩에서 처리해야 할 프레임을 많이 줄일 수 있다. (그림 13)는 예측주기가 3일 때의 경우를 설명하고 있다. (그림 12)의 경우와 마찬가지로 설명할 수 있고, 각각의 경우에 따른 처리가 필요한 프레임의 개수는 (표 3)와 같다. (표 3)에는 예측주기와 삭제주기에 따라 처리할 프레임의 개수가 비교되어 있고 예측주기와 삭제주기의 2, 3, 그리고 4의 최소공배수인 전체 12 프레임을 기준으로 계산되었다.

(그림 14)은 (그림12, 13)에서 설명된 동적 프레임율 조절을 위한 Processing Frame Selection 알고리즘이다.

표 3. 예측주기 및 삭제주기에 따른 처리할 프레임율의 비교

삭제 주기 \ 예측 주기	2	3	4
2(2프레임당 하나의 프레임 삭제)	0%	33%	0%
3(3프레임당 두 개의 프레임 삭제)	67%	0%	42%
4(4프레임당 세 개의 프레임 삭제)	50%	50%	0%

3.3 트랜스코딩 여부를 판별하는 클라이언트의 복호기

대부분의 사람들이 사용하고 있는 MicrosoftTM의 Windows에 기본적으로 설치되어 있는 Window Media Player, 스트리밍 서비스로 유명한 RealNetworksTM 의 RealPlayer, 그리고 AppleTM의 QuickTime은 전용 재생기를 제공하고 있다. 이와 같은 전용 재생기는 간단한 플러그인의 설치만으로 비디오 스트리밍 서비스에 많이 사용된다.

본 논문에서 제안한 서버의 부호기로 변경된 비디오 스트림은 트랜스코딩을 거치지 않으면 일반 클라이언트에서 제대로 재생되지 않을 수도 있다. 그래서 (그림 15)와 같이 하나의 스위치를 추가하여 트랜스코딩의 여부에 따라 올바로 재생될 수 있도록 제안하였다. 일반적인 복호기에 비해 큰 처리 비용을 필요로 하지 않기 때문에 열악한 환경의 클라이언트에도 어렵지 않게 사용될 수 있다.

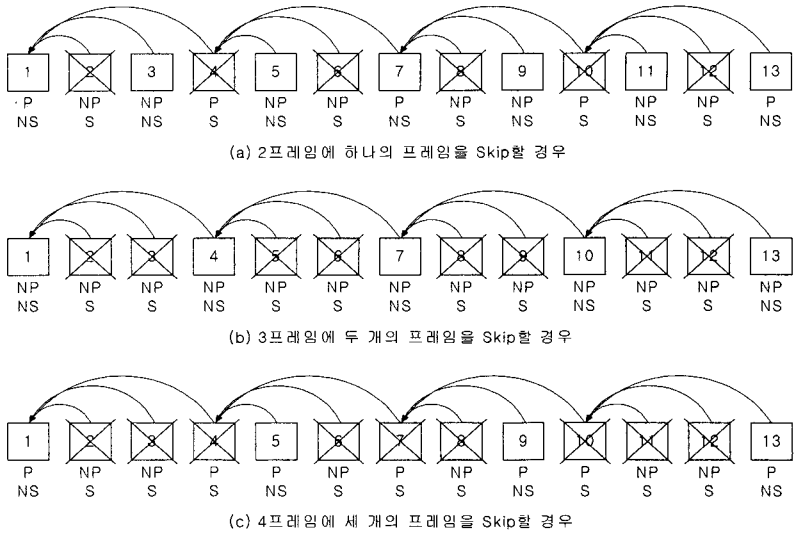


그림 13. 예측주기가 3인 프레임을 제거할 경우 : (a) 2프레임에 하나의 프레임을 Skip할 경우, (b) 3프레임에 두 개의 프레임을 Skip할 경우, (c) 4프레임에 세 개의 프레임을 Skip할 경우.

```
Procedure Processing_Frame_Selection()
Begin Procedure              // 요청된 프레임율에 따라 삭제주기 결정
nsp = 삭제 주기
pp = 예측 주기
If (pp == nsp * alpha)       // alpha = 1, 2, 3,
  Exit Procedure             // Bypass의 경우
End If
lcm = LCM(pp, nsp)           // pp와 nsp의 최소공배수
last_pp_skip=1               // 마지막 예측주기 프레임의 삭제 여부
                             // 1: 삭제된 프레임, 0: 남은 프레임
k = 1                        // 프레임 번호
For ( k <= lcm )
  If((k-1) % pp == 0)
    Processing Frame         // 처리할 프레임
  Else
    If(pp <nsp)              // 삭제주기 > 예측주기
      If((k-1) % nsp == 0)
        Processing Frame     // 처리할 프레임
      End If
    Else                     // 삭제주기 < 예측주기
      If(last_pp_skip == 1 && (k-1) % nsp == 0)
        Processing Frame     // 처리할 프레임
      End If
    End If
  End If
  If((k-1) % pp==0 && (k-1) % nsp==0) // 마지막 예측주기 프레임의 삭제 여부
    last_pp_skip = 0
  End If
  k++;                       // 다음 프레임
End For
End Procedure
```

그림 14. Processing Frame Selection 알고리즘

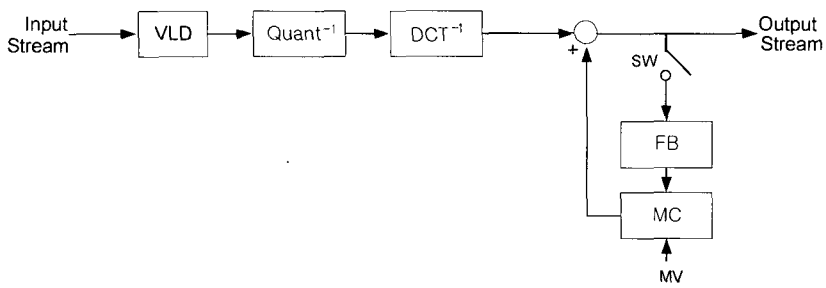


그림 15. 클라이언트 복호기

4. 성능 분석

제안된 스트리밍 시스템의 성능 평가를 위해서 여러 가지 실험 인자를 통해 전통적인 동적 프레임 삭제 기법[17]과 처리시간 및 출력 스트림의 화질을 비교하였다.

실험에 사용된 비디오 스트림은 QCIF 포맷의 H.263[6, 20] 파일이고 "IPPP..." 의 구조를 가진다. 성능 평가를 위해 표준 비디오인 Suzie와 Football을 사용하였으며, Suzie와 Football은 전체 스트림에서 각각 44%와 82%의 0 아닌 움직임 벡터를 포함하고 있다. 실험에 사용된 부호기 및 트랜스코더는 FFmpeg[7]의 H.263 코덱을 수정하여 구현하였다. (표 4)와 (표 5)에는 각각 입력 비디오 스트림과 실험 환경이 나타나 있다.

표 4. 입력 비디오 스트림

시퀀스	양자화 파라미터	Video Format	프레임율 (fps)	프레임 개수
Football	5	QCIF	30	257
Suzie				149

표 5. 실험 환경

시스템 환경	CPU & 메모리	Pentium IV 1.2 Ghz CPU, 256M
	운영체제	Redhat Linux 9.0
비디오 타입	H.263	"IPPP.." 시퀀스
비디오 코덱	FFmpeg	Version 0.4.8
출력 프레임율	15, 10, 7.5	삭제 주기를 2~4로 조절
출력 양자화 인수	5	

(그림 16,17)은 [17]에 제안된 트랜스코더와 본 논문에서 제안한 트랜스코더의 예측주기가 2, 3, 4일 때, 삭제주기가 2인 경우의 움직임이 많은 스트림에 대한 계산 복잡도(μs)와 출력화질(dB)을 나타내고 있다.

(표 6)에는 두 개의 비디오 스트림에서 기존의 트랜스코더와 제안된 트랜스코더의 평균 PSNR과 처리시간이 나타나 있다. 실험을 통한 결과에서 제안된 스트리밍 시스템은 기존의 방법에 비해 비슷한 출력 화질과 약 8%~65%정도의 계산 복잡도로 향상된 성능을 보였다.

(그림 18, 19)은 예측주기별 출력 스트림의 특정 프레임별 화질을 비교하여 나타내고 있다. 삭제주기가 2일때 기존의 트랜스코더를 거친 프레임과 제안

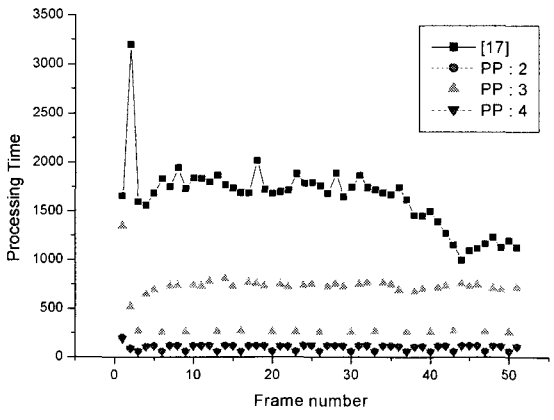


그림 16. football 스트림에서 예측주기별 계산복잡도

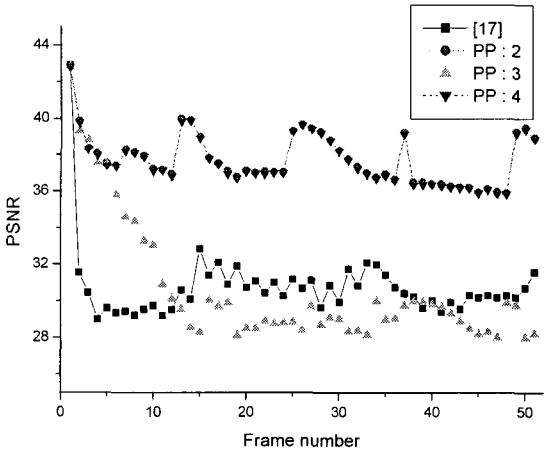


그림 17. football 스트림에서 예측주기별 화질

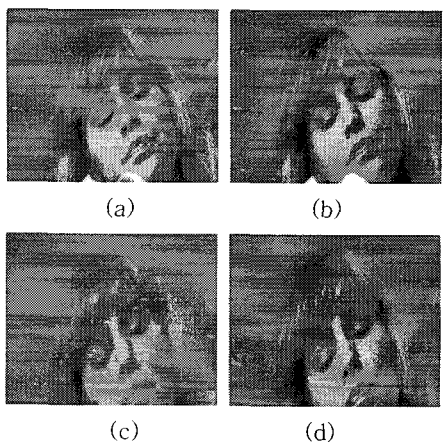
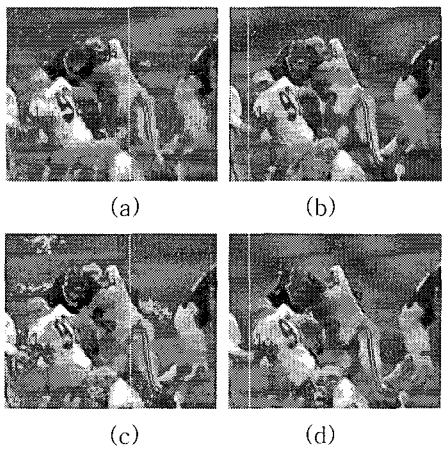


그림 18. suzie 스트림에서 예측주기별 화질 비교(72번째 프레임)



(a) (b)

(c) (d)

그림 19. football 스트림에서 예측주기별 화질 비교(24번째 프레임)

된 트랜스코더를 거친 프레임이 나타나 있다. 각 그림별 (a)는 기존의 트랜스코더에서의 프레임을 나타내고, (b)~(d)는 제안된 트랜스코더에서 예측주기가 2, 3, 4일 때의 결과 화면이다.

앞서 제시된 표와 그림을 통해서 제안된 트랜스코더는 [17]에서 제안된 트랜스코더에 비해 비슷하거나 조금 더 나은 PSNR을 보인다. 또한 처리시간은 예측주기가 삭제주기의 정수배인 특별한 경우를 제외하더라도 전체적으로 절반가량의 시간을 필요로 한다. 그러므로 처리시간에서 기존 기법보다 매우 우수한 성능을 보임을 확인할 수 있다.

본 논문에서 제안한 트랜스코딩은 예측주기를 가지는 스트림에 대해서 활용가능하다. 따라서 예측 주기를 사용한 부호기 측의 효율성과 활용가능성을 확인하기 위하여 예측주기가 없는 기존의 부호기와 예측주기가 있는 부호기의 성능을 비교하였다. 성능 인자는 부호기를 통해 생성된 스트림의 프레임별 평균 출력화질(PSNR)과 평균 처리시간(μs)을 비교하였다.

표 7은 실험결과를 나타내고 있다. 예측주기를 통해 예상되었던 출력화질의 열화는 극히 미미하게 나타났고, 이에 반해 처리 시간 향상의 폭은 크게 나타났다. 따라서 예측주기를 이용한 부호화기는 일반적인 미디어 전송을 위해서도 활용가능하며, 실시간성을 높이 요구하는 화상전화, 화상회의 시스템, 실시간 방송 서비스에도 활용가능하다.

표 7. 제안부호기와 일반부호기의 성능 비교

비교항목		football	suzie
일반 부호기	평균 PSNR	38.63	39.91
	처리시간(μs)	775348	594388
제안 부호기 (예측주기:2)	평균 PSNR	37.88	39.66
	처리시간(μs)	697813	534949
제안 부호기 (예측주기:3)	평균 PSNR	37.80	39.10
	처리시간(μs)	682306	523061
제안 부호기 (예측주기:4)	평균 PSNR	37.72	39.02
	처리시간(μs)	666799	511174

표 6. 트랜스코더의 예측주기와 삭제주기별 성능 비교

비교항목			football	suzie
기존트랜스코더 (삭제주기:2)		평균 PSNR	30.72	38.21
		처리시간(μs)	198985	71786
예측주기:2	삭제주기:2	평균 PSNR	37.88	39.66
		처리시간(μs)	11481	4735
	삭제주기:3	평균 PSNR	30.18	37.23
		처리시간(μs)	131149	46958
	삭제주기:4	평균 PSNR	30.01	36.53
		처리시간(μs)	117645	38997
예측주기:3	삭제주기:2	평균 PSNR	30.94	36.29
		처리시간(μs)	67782	25182
	삭제주기:3	평균 PSNR	37.80	39.10
		처리시간(μs)	13039	4745
	삭제주기:4	평균 PSNR	29.83	36.08
		처리시간(μs)	101618	37217
예측주기:4	삭제주기:2	평균 PSNR	37.72	39.02
		처리시간(μs)	11223	4958
	삭제주기:3	평균 PSNR	29.68	34.38
		처리시간(μs)	83198	31198
	삭제주기:4	평균 PSNR	37.72	39.02
		처리시간(μs)	11739	4435

5. 결론 및 향후과제

본 논문에서는 동적으로 프레임율 조절이 가능한 스트리밍 시스템을 위하여 트랜스코딩 기법을 제안하였다. 프레임율을 조절하는 트랜스코딩에서 계산 복잡도를 최소화하기 위해서 서버의 부호기에서 예측주기를 이용하였다. 트랜스코더는 기존에 제시된 트랜스코더[17] 보다 개선된 움직임 예측을 수행하기위해 움직임 합성 기법을 적용하였으며, 삭제주기에 따라 기존의 트랜스코더에 비해 처리시간을 줄이도록 구성하였다.

트랜스코더에 입력되는 비디오 스트림은 예측주기가 적용되어 프레임율 조절이 용이한 구조를 가지므로 프레임율 조절 시에 기존의 트랜스코더 구조보다 평균 60%이상의 처리 시간을 감소 시켰다.

본 논문에서 제안하는 트랜스코딩 기법은 단방향 예측의 비디오 스트림에 적용 가능하다. 따라서, 향후에는 양방향 예측이 있는 비디오 스트림에서도 프레임율을 효율적으로 조절할 수 있는 연구가 필요하다. 또한, 다양한 실험을 통해 컨텐츠의 종류나 그에 따른 움직임 특성에 따라서 적절한 예측 주기를 제공할 수 있는 향후 연구가 필요하다.

참 고 문 헌

[1] M. S. Chen and D. D. Kandlur, "Downloading and stream conversion: Supporting interactive playout of videos in a client station," *in Proc. 2nd International IEEE Conference of Multimedia Computing and Systems*, pp. 73-80, 1995.

[2] T. D. C. Little and D. Venkatesh, "Prospects for interactive video-ondemand," *IEEE Trans. Multimedia*, Vol. 13, pp. 14-24, Aug. 1994.

[3] B. Shen, I. K. Sethi, and B. Vasudev, "Adaptive motion vector resampling for compressed video downscaling," *IEEE Trans. on Circuits and Systems for Video Technology*, Vol. 9, pp. 929-936, Sept. 1999.

[4] V. Bhaskaran and K. Konstantinides, *Image and Video Compression Standards*, Kluwer Adademic Publishers, 1994.

[5] N. Bjrk and C. Christopoulos, "Transcoder architectures for video coding," *IEEE Trans. Consumer Electronics*. Vol. 44, pp. 88-98, Feb. 1998.

[6] International Telecommunications Union, "Video coding for low bitrate communication," *Geneva, Switzerland, ITU-T Recommendation H.263*, 1998.

[7] FFmpeg multimedia system. [Online]. Available: http://www.ffmpeg.org.

[8] A. Vetro, C. Christopoulos, and H. Sun, "Video Transcoding Architectures and Techniques: An Overview," *IEEE signal Processing Magazine*, Vol. 20, No. 2, pp. 18-29, March 2003.

[9] T. Warabino, S. Ota, D. Morikawa, M. Ohashi, H. Nakamura, H. Iwashita, and F. Watanabe, "Video Transcoding Proxy for 3Gwireless Mobile Internet Access," *IEEE Communications Magazine*, Vol. 38, No. 10, pp. 66-71, Oct. 2000.

[10] N. Bjork and C. Christopoulos, "Video Transcoding for Universal multimedia Access," *ACM Workshop on Multimedia Standards, Interoperability and Practice (MM2000 Workshop), in Proc. ACM Multimedia 2000*, pp. 75-79, Nov. 2000.

[11] T. shanableh and M. Ghanbari, "Heterogeneous Video Transcoding to Lower Spatio-Temporal Resolutions and Different Encoding Formats," *IEEE Trans. Multimedia*, Vol. 2, No. 2, pp. 101-110, June 2000.

[12] Y. Nakajima and M. Sugano, "MPEG bit rate and format conversions for heteroheneous network/storage applications," *IEICE Trans. Electronic*, E85-C, No. 3, pp. 492-503, Mar. 2002.

[13] M. Emad Modirzadeh, "Bit-Rate Reduction of MPEG Compressed Video," *in Proc. IEEE Canadian Conference on Electrical and Computer Engineering*, Vol. 2, pp. 1049-1054, 2002.

[14] H. Sorial, W. Lynch, and A. Vincent, "Selective Requantization for Transcoding of MPEG Compressed Video," *in Proc. IEEE International Conference on Multimedia and Expo (ICME 2000)*, NY, USA, Vol. 1 pp. 217-220, 30 July - 2 Aug. 2000.

[15] J. Youn, M.T. Sun, and J. Xin, "Video Transcoder Architectures for Bit Rate Scaling of H.263 Bit Streams," *in Proc. ACM Multimedia'99*, pp. 243-250, Nov. 1999.

[16] H.Sun, W. Kwok, and J. W. Zdepski, "Architecture for MPEG compressed bitstream scaling," *IEEE Trans. on Circuits and Systems for Video Technology*, Vol. 6, pp. 191-199. Apr. 1996.

[17] K. T. Fung, Y. L. Chan, and W. C. Siu, "New architecture for dynamic frame-skipping transcoder," *in Proc. IEEE Workshop Multimedia signal Processing*, Redondo Beach, CA, pp. 616-621, Dec. 1998.

[18] M. J. Chen, M. C. Chu, and C. W. Pan, "Efficient motion estimation algorithm for reduced frame-rate video transcoder," *IEEE Trans. on Circuits and Systems for Video Technology*, Vol. 12, pp. 269-275, Apr. 2002.

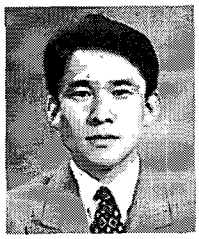
[19] J. Youn, M. T. Sun, and C. W. Lin, "Motion vector refinement for high performance transcoding," *IEEE Trans. Multimedia*, Vol. 1, pp. 30-40, Mar. 1999.

[20] Guy Côté, Berna Erol, Michael Gallant, and Faouzi Kossentini, "H.263+: Video coding at Low Bit Rates," *IEEE Trans. on Circuits and Systems for Video Technology*, Vol. 8, pp. 849-866. Nov. 1998.

김 성 민

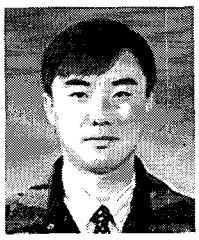
2001년 부산대학교 전자계산학과 졸업
2003년 부산대학교 전자계산학과 석사
2003년~현재 부산대학교 컴퓨터 공학과 박사과정

관심분야 : 트랜스코딩, 멀티미디어 스트리밍, 유비쿼터스 컴퓨팅



김 영 주

1988년 부산대학교 계산통계학과 (이학사)
1990년 부산대학교 계산통계학과 (이학석사)
1990년~1995년 큐닉스컴퓨터 응용시스템연구소
1999년 부산대학교 전자계산학과 (이학박사)
2000년~현재 신라대학교 컴퓨터공학과 조교수
관심분야 : 임베디드시스템, 멀티미디어처리, 영상 압축 및 통신



박 성 호

1996년 부산대학교 전자계산학과 졸업 (학사)
1998년 부산대학교 대학원 전자계산학과 졸업(이학석사)
2002년 부산대학교 대학원 전자계산학과 졸업(이학박사)
2002년 9월~현재 부산대학교 정보전산원 조교수
관심분야 : VOD 시스템, 인터넷 캐슁, 멀티미디어 통신, 비디오 트랜스코딩, 임베디드시스템