MPEG-1 비디오 스트림에 대한 압축 영역에서의 장면 전환 효과 처리

(Shot Transition Effects for MPEG - 1 Video Stream in Compressed Domain)

이 승 철 † 낭 종 호 ††
(Seungcheol Lee) (Jongho Nang)

요 약 MPEG 형식의 비디오 데이타의 사용이 점차 늘어남에 따라, 이런 데이타를 자유롭게 편집할 수 있는 편집기의 필요성이 점차 증가하고 있다. 그러나 MPEG 데이타는 연속된 프레임들의 차이만을 저장하는 방법을 사용하기 때문에 두 개의 MPEG 데이타를 전환 효과를 주어서 편집을 하기 위해서는 압축을 해제한 후 효과를 주고 다시 압축하여야 하는 문제점을 가지고 있다. 이런 문제점을 해결하기 위하여 본 논문에서는 MPEG 형식으로 압축된 두 비디오 데이타에 대하여 부분적으로 압축을 해제한 상태에서 여러 종류의 장면 전환 효과(페이드 인 및 페이드 아웃, 디절브)를 적용할 수 있는 방법을 제안하였다. 제안한 방법에서는 MPEG형식 비디오 데이타의 I와 P프레임에 대하여서는 압축을 부분적으로 해제하는 전환 효과를 주는 기존의 방법을 응용하여 적용하였으며, B 프레임의 경우에는 참조 프레임에 대한 움직임 벡터를 기초로 구해진 움직임 보상 근사값을 이용하여 DCT 영역에서 장면 전환 효과를 주는 방법을 새로이 제안하였다. 이런 압축 영역에서의 장면 변화 적용 방법을 사용함으로써 압축을 해제하여 효과를 주는 것 보다 전환되는 프레임들의 화질을 유지하면서 3-4배 정도 빠르게 편집할 수 있다. 이러한 방법을 MPEG 비디오 편집기 시스템에 적용하게 된다면, 소프트웨어만을 이용한 저가의 데스크 탑 환경에서도 효율적인 비디오 편집 작업이 가능하게 될 것이다.

Abstract As the full-motion videos in MPEG are widely available nowadays, an editor that could easily edit such kind of media data is required to develop various multimedia applications. In order to concatenate and apply a transition effect to two video streams encoded in MPEG, they should be decoded first since there are dependencies in the frames in MPEG-encoded video stream. Since this decode-edit-encode process requires a huge amount of computing/storage resources, a new editing scheme that could apply various transition effects to MPEG video streams directly while keeping the quality of video data is strongly required. This paper proposes a new editing scheme that could apply three transition effects (such as fade-in, fade-out, and dissolve) to MPEG video streams in a compressed domain. In the proposed scheme, an extension of previous method in which the frames are partially decompressed and transition effects are applied is adopted for I- and P-frames. In addition, a new processing scheme for B-frame that could apply the transition effects in DCT domain directly using an approximation of motion compensation based on the motion vector to reference frames. Since this processing scheme could apply the transition effects in a compressed domain directly, the editing process could be speed-up about 3~4 times faster than previous decode-edit-encoding method while keeping the quality of video data as good as the source data. The proposed scheme could be used to build a software-only MPEG video editing system that helps to edit MPEG video data even on a low-cost desk-top computer.

· 본 연구는 한국과학재단 산학협력 연구 (제목 : MPEG 비디오 스트림에 대한 비선형 편집기 개발)의 연구비 지원에 의하여 이루어졌음.
† 비 회 원 : 서강대학교 컴퓨터학과
leesc@chunghocom.co.kr
†† 종신회원 : 서강대학교 컴퓨터학과 교수
jhnang@ccs.sogang.ac.kr
논문접수 : 1999년 6월 11일
심사완료 : 2000년 3월 8일

1. 서 론

MPEG 비디오 형식의 데이타는 디지털 동영상을 매우 높은 비율로 압축하여 저장하고 전송할 수 있기 때문에 많은 멀티미디어 분야에서 사용되고 있다. 이들 비디오 스트림을 원하는 목적에 알맞게 가공하기 위해서 편집기가 필요하며, 압축된 데이타의 특성상 편집기는 고성능 컴퓨터 및 매체를 디코딩 및 인코딩할 수 있는 코덱을 구현한 추가적인 하드웨어가 필요하다. 하지만, 요즈음 CPU의 성능이 급속도로 발전하여 개인용 컴퓨터에서 구동되는 소프트웨어 형태의 간단한 탁상용 편집기 시스템만으로 자유로운 편집 및 다양한 편집 효과를 구가할 수 있게 되었다. 한편, 이러한 소프트웨어 편집기를 이용하여 편집을 하고, 각종 편집효과를 가하게 되면 원본 비디오 스트림보다 다소 화질이 떨어지는 출력물을 얻게 되며, 또한 이를 수행하는 시간도 너무 오래 걸리는 단점을 안고 있다. 이러한 문제점이 발생되는 원인은 편집 효과를 압축된 비디오 스트림을 원시 형태로 완전히 복원한 후에 작용하게 되고, 이를 다시 인코딩하는 방법을 사용하기 때문이다.

원시 형태의 영상을 MPEG으로 인코딩하는 여러 단계의 과정 중에서 많은 시간을 차지하는 것은 이산 코사인 변환과 움직임 예측이다[7]. 또한 인코딩한 영상을 다시 디코딩 하였을 때에 원래의 영상과 차이가 나는 이유는 양자화 과정을 거치기 때문이다[6, 7]. 전술한 편집 효과의 처리 방법에서 노출된 문제점을 해결하기 위해서는 많은 시간을 요구하는 과정과 화질 저하를 유발하는 과정을 거치지 않고 편집 할 수 있는 방법을 찾아야 한다. 이런 MPEG 데이타의 편집에 관한 기존의 연구 방법들을 살펴보면, 빠른 수행 시간을 얻기 위하여 이산 코사인 변환 영역에서 여러 가지 특수 효과를 처리하는 방법[1, 2, 3]이 제안되었으며, 구현하려는 효과의 특성에 맞게 최적화된 움직임 보상 기법 및 이산 코사인 변환 영역에서의 블록 내 연산법을 사용하는 방법[16, 17] 등이 제안되었다. 그러나 이런 방법들은 대부분 단일 스트림에 대한 특수 효과나 부분적인 프레임간의 합성[1, 3] 등을 처리하기 위해 제안되었기 때문에, 2개 이상의 MPEG 비디오 스트림에 대하여 장면 전환 효과를 필요로 하는 편집에는 사용될 수 없다.

본 논문에서는 MPEG 형식의 비디오 데이타에 대하여 편집기에서 가장 많이 사용되는 전환 효과인 페이드 인 및 페이드 아웃(fade in & fade out)과 디졸브(dissolve) 효과를 압축 영역에서 처리할 수 있는 새로운 방법에 대하여 제안한다. 모든 효과는 I 프레임의 경우 이산 코사인 변환 영역에서 적용되며, P 프레임은 원시 데이타 형태까지 복원하는 과정을 거치고 다시 이산 코사인 변환 과정을 통하여 생성이 된다. 그리고 B 프레임의 경우에는 움직임 보상 근사처리 방법을 사용하기 때문에, 이산 코사인 변환 영역까지만 디코딩 될 뿐 아니라, 움직임 예측 과정도 거치지 않게 된다. 전환 효과가 가지는 특성, 즉 두 개의 영상의 합성이 이루어지는 점을 살려서, 불필요하게 정밀한 처리 과정을 간략화하고 처리하는 스트림 내에서 사용하는 움직임 벡터를 그대로 사용하는 방법을 제안하였다. 본 논문에서 제안한 방법을 실제 구현하여 실험한 결과에 의하여 MPEG형식 비디오 데이타와 화질을 저하하기 않으면서 3~4배 정도의 처리 속도 향상을 얻을 수 있음을 알 수 있었다.

본 논문에서는 제 2 장에서 기존의 전환 효과 처리 방법에 대하여 살펴보도록 한다. 제 3 장에서는 본 논문에서 제안하는 장면 전환 효과 처리 방법에 대하여 구체적으로 설명할 것이며, 제 4 장에서는 제안한 방법과 기존의 전환 효과 처리 방법을 실제로 구현해 보고, 예제 스트림을 통하여 비교 실험 및 그 결과 분석을 통한 성능 평가를 하도록 하겠다. 제 5 장에서는 성능 평가 결과를 토대로 관련 연구와의 비교 결과를 설명하며, 제 6 장에서 결론을 내리고 추후 연구 방향을 제시하였다.

2. 기존의 전환 효과 처리 방법

MPEG 형식의 비디오 스트림을 대상으로 하는 소프트웨어 편집기 시스템은 디지털 미디어가 가지는 편리함 및 유용성으로 인해 많이 개발되어 왔고, 시중에는 상용화 된 제품도 많이 나와있다. 이들 제품에서 가장 핵심이 되는 요소는 잘라 붙이기(Cut & Paste) 기능과, 전환 효과(Transition effect)라 할 수 있다. 이는 많은 대상 스트림들을 의미가 있는 장면들을 잘라서 붙이고, 그 경계에 대하여 특수효과를 가하는 것이 편집기의 주된 목적이기 때문이다. 디지털 미디어에 대하여 편집이 이루어지는 과정을 살펴보면, 잘라 붙이기 작업 후에 전환 효과를 가하는 작업이 주로 이루어지게 된다. 두 개의 비디오 스크림을 붙이는데 사용되는 전환효과로는 다음과 같은 것이 있다[11].

- 페이드(fade) : 하나의 입력 신호에서 다른 검정색과 같은 일정한 배경으로 전환을 하는 데 점진적으로 입력 신호가 줄어들면서 다른 배경으로 바뀌는 효과이다. 점차 어두워지는 경우를 페이드 아웃, 그 반대의 경우를 페이드 인이라고 한다.
- 디졸브(dissolve) : 두 개의 입력을 받아서 각 신호에 가중치를 두면서 더한 신호를 출력받는 것이

다. 한 입력 신호는 점점 감소하고, 다른 입력 신호는 점차 증가하게 된다. 이로써 자연스럽게 장면이 바뀌게 된다.

- 와이프(wipe) : 한 입력 신호가 다른 입력 신호에 의해서 스크롤 되면서 쓸려 나가는 식으로 전환되는 효과이다.
- 래핑(wrapping) : 평면적인 화면이 원이나 다면체 모양과 같은 3차원 개체의 표면으로 매핑되어 보여지는 효과이다.

본 절에서는 이런 전환 효과를 처리할 수 있는 기존의 여러 가지 방법들을 소개할 것이다. 일반적으로 생각해 볼 수 있는 공간 영역에서의 전환 효과 처리 방법에 대하여 설명하고, 이 방법이 가지는 여러 가지 문제점을 지적한 후, 이러한 문제점을 해결하기 위한 기존의 여러 가지 연구에 대하여 소개를 할 것이다.

2.1 공간 영역(Spatial Domain)에서의 전환 효과 처리

MPEG 형식으로 압축된 비디오 데이타에 대하여 앞에서 설명한 전환 효과를 줄 수 있는 가장 쉬운 편집 방법은 압축을 해제하여 원시 데이타(Raw Data)를 만들고, 이런 데이타에 전환 효과를 적용하고, 그 결과를 다시 MPEG 형식으로 압축하는 것이다. 원시 데이타에 대하여 전환 효과를 주는 방법을 좀 더 자세하게 설명하면 다음과 같다.

(1) 페이드 인 및 페이드 아웃(Fade in & fade out)

먼저 페이드 인 및 페이드 아웃을 생각해 보자. 이는 비디오 스트림의 현재 장면과 대상이 되는 정지 영상과의 가중치 평균한 값을 기록하게 되며, 가중치 값은 효과가 가해지는 시간 동안에 일정하게 커지거나 작아지게 된다. 이를 수식으로 표현하게 되면, 식(1)과 같다[12]. 여기서 $P_a(i,j)$는 현재의 장면에서 좌표(i, j)의 화소값, $P_b(i,j)$는 바뀌어져야할 장면에서 좌표(i, j)의 화소값, α는 가중치, $P_{new}(i,j)$는 새롭게 생성되는 장면에서 좌표(i, j)의 화소값을 나타낸다. <그림 1>-(a)에서 보는 바와 같이 페이드 인의 경우, 가중치 α값은 0에서 1로 서서히 증가하게 되며, 페이드 아웃(<그림 1>-(b))의 경우 그 반대가 될 것이다. 그리고, P_b는 검정색의 정지 영상이 된다. 이 식에서 t는 전환 효과가 가해지는 구간이 되며, 이때 α값은 $\frac{1}{t}$만큼씩 변하게 된다.

(2) 디절브(Dissolve)

디절브 효과를 공간 영역에 대하여 처리하기 위해서도 위의 식(1)이 그대로 적용된다. 하지만 이 경우에는 P_a, P_b 모두 MPEG 비디오 스트림 속의 하나의 장면

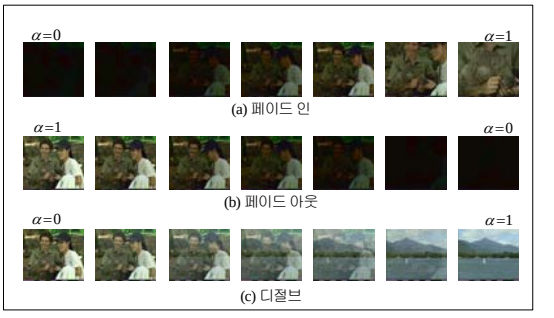


그림 1 대표적인 전환 효과들

이 된다. <그림 1>-(c)에서 보는 바와 같이 시간의 흐름에 따라 장면 서서히 겹쳐지면서, 가중치 α값은 1에서 0으로 서서히 감소하게 된다.

$$P^i_{new}(i,\ j) = \alpha \cdot P^i_a(i,\ j) + (1-\alpha) \cdot P^i_b(i,\ j) \quad (1)$$
$$0 \le \alpha \le 1$$

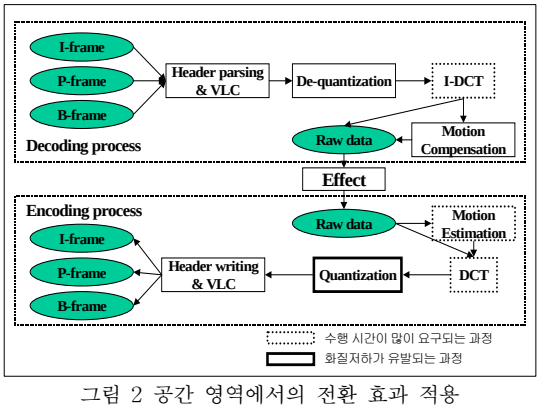


그림 2 공간 영역에서의 전환 효과 적용

MPEG 비디오 스트림은 원시 형태의 영상에 비하여 수십 배의 압축효과[6, 7]를 얻을 수 있지만, 디코딩하고 인코딩하는 데에 많은 계산을 요구하게 된다. <그림 2>는 공간 영역에서 이런 전환 효과를 처리하기 위한 단계를 도식화한 것이다. 이 그림은 MPEG 1 비디오 디코더의 출력과 인코더의 입력 부분을 연결시켜 놓은 것으로, 핵심이 되는 단계들을 나타내었으며, 화질의 저하를 유발하는 부분과 상대적으로 많은 연산 수행 시간을 요구하는 과정을 별도로 표시해 놓았다. 전환 효과를 가하기 위해서 압축된 데이타를 원시 형태로 완전히 복원하는 과정을 살펴보면, IDCT 및 움직임 보상 단계를 거치게 된다. 또한 효과를 가한 후에 다시 인코딩 하는 과정에서 DCT 및 움직임 예측과 양자화를 거치게 되는

데, 이러한 일련의 과정은 많은 수행 시간을 필요로 한다. 이 방법은 전환 효과 방법 그 자체를 구현하기는 쉽지만, 하드웨어의 추가 없이 소프트웨어로 구현되는 비디오 편집기의 경우 긴 작업시간과 화질의 저하로 인해 효율적이지 못하다는 문제점을 가지고 있다.

2.2 DCT 영역에서의 전환 효과

공간 영역에서의 효과처리에서 나타난 비효율성을 개선하기 위하여 많은 연구가 이루어져 왔으며, 그 해결 방안으로써 DCT 영역에서 특수 효과를 처리하는 방법[1, 2, 3, 16, 17]이 많이 제안되었다. 이 처리 기법의 기본 원리는 식(2)에 나타낸 DCT 영역 계산의 특성을 바탕으로 한다. 즉, DCT 영역에서의 실수의 곱셈 및 덧셈에 의해 공간 영역에서 얻어낼 수 있는 값과 동일한 값을 얻어 낼 수 있다는 성질을 이용하는 것이다.

$$DCT(P_{new}) = \alpha \cdot DCT(P_a) + (1-\alpha) \cdot DCT(P_b) \qquad (2)$$

식(2)는 식(1)과 동치이다. 즉, 식(2)에서 새롭게 얻게 되는 블록 내 화소값들은 식(1)에 의해 얻게되는 식을 DCT하게 되면 동일한 값이 된다. 물론 그 역과정 역시 마찬가지가 된다. 따라서, 페이드 인 및 페이드 아웃 효과와 디졸브 효과 역시, DCT 영역에서 그대로 처리하여도 동일한 결과를 얻을 수 있게 된다. 이렇게 DCT 영역에서 페이드 인 및 페이드 아웃과 디졸브 효과를 처리하기 위해서는 특별히 I 프레임으로만 구성된 MPEG 비디오 스트림을 생성해 낼 수 있는 MPEG 인코더가 필요하게 된다. <그림 3>은 이러한 과정을 도식화 한 것으로, I 프레임은 IDCT를 거치지 않고, 직접 전환 효과 과정을 거치게 되고, 나머지 P, B 프레임은 완전히 원시 데이타로 복원된 상태까지 디코딩된 후, 다시 DCT를 거친 후에 전환 효과 과정을 거쳐 최종적으로 I 프레임만으로 구성된 MPEG 비디오 스트림을 출력하게 된다. 공간 영역에서 효과를 처리하는 방법과의 차이점은 B, P 프레임을 생성하지 않기 때문에, 수행시간이 오래 걸리는 작업인 움직임 예측 과정을 생략할 수 있다는 점과, I-프레임 만으로만 구성된 MPEG 비디오 스트림이 그 대상으로 될 경우 처리 속도가 훨씬 빨라지게 된다.

한편, <그림 3>에서 설명한 효과 방법은 움직임 보상을 원시 형태의 데이타에 대하여 수행하고 있지만, 이를 DCT 영역에서 가능하도록 하게 되면, IDCT를 거치지 않아도 된다. 이를 위하여 블록 추출 및 재구성 기법[2]을 응용하여 적용할 수 있다. 하지만 이는 P, B 프레임 내에서 인트라 형식의 블록의 개수가 많은 경우에 성능 개선 효과가 있지만, 일반적인 경우인 많은 수의 매크로블록이 참조 형식으로 된 경우에는 좋은 효과를 기대하기 힘들다[1, 3]. 또한, DCT 영역에서 블록 내 연산 기법[16, 17]을 사용하는 방법도 있다. 이 방법은 I 프레임 뿐만 아니라 B, P 프레임에 대해서 DCT 영역에서 특수 효과를 적용할 수 있지만, 두 개의 스트림에 대하여 효과를 가해야 하는 전환 효과에 적용할 수 있는 기법은 아니다.

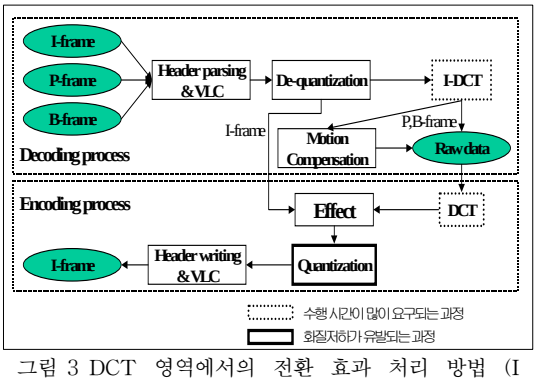


그림 3 DCT 영역에서의 전환 효과 처리 방법 (I frame only stream을 생성)

3. 움직임 보상 근사처리에 기초한 전환 효과 처리 방법

일반적으로 MPEG 비디오 데이타를 디코딩 단계 및 인코딩 단계에서 특정 단계를 생략하거나, 근사 처리하게 될 경우 빠른 수행시간을 얻을 수는 있지만, 이로 인하여 계산 오차가 발생하게 되고 이는 화질이 좋지 못한 영상을 출력하게 되는 결과를 낳게된다. 하지만, 전환 효과라는 특별한 조건이 주어진 상황에서는 디코딩 및 인코딩의 각 단계에서 굳이 실행하지 않아도 될 것과 근사 처리해도 크게 문제될 것이 없는 요소들이 나타날 가능성이 있다. 본 논문에서는 이러한 특성을 이용하여 MPEG 형식으로 인코딩된 동영상에 대하여 I 프레임의 디코딩과 P 프레임의 디코딩 및 인코딩, B 프레임에서의 움직임 보상 근사처리에 의한 페이드 인 및 페이드 아웃과 디졸브 효과를 처리할 수 있는 새로운 방법을 제안하겠다. 이 방법에서는 B 프레임내 움직임 보상이 일어나게 되는 매크로블록에 대하여 근사 처리법을 적용 시켜 DCT 영역에서 전환 효과에 의한 그림의 변화를 기록할 수 있게 된다. <그림 4>은 제안한 방법의 처리과정을 전체적으로 나타낸 것이다. 페이드 인 및 페이드 아웃 효과의 경우 기본적인 효과 처리 과

정의 두 입력 스트림 중 하나를 검정색 단위 블록으로 대치하게 되며, 디졸브 효과의 경우 효과 처리 방법 이외에 별도의 루틴을 두어 화질 저하 구간에 대한 처리를 하도록 설계되었다.

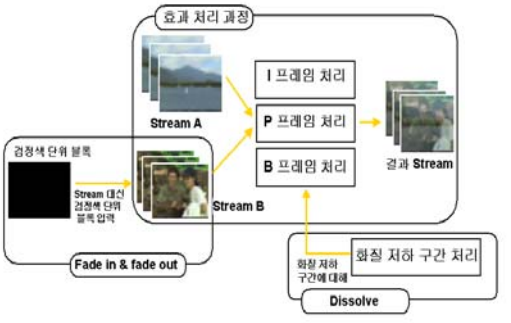


그림 4 전체 전환 효과 처리 과정

1절에서는 상이한 두 개의 그림에서의 겹치기(Overlap) 효과를 가할 수 있는 효과 처리 방법에 대하여 설명하고, 2절 및 3절에서는 각각 페이드 인 및 페이드 아웃과 디졸브 효과에 대한 구체적인 방법을 제시할 것이다.

3.1 효과 처리 방법

이 절에서 설명할 내용은 대표적인 장면 전환 효과인 페이드 인 및 페이드 아웃과 디졸브에 대한 처리 방법에 관한 것으로, 공통적으로 적용되는 효과 처리 방법이다. MPEG 비디오 스트림에 대하여 각 프레임별로 해당하는 처리 방법에 대하여 설명하겠다.

3.1.1 I 프레임의 처리

우선 I 프레임에 대한 처리에 대하여 설명하도록 한다. 이 처리 방법에서는 <그림 5>와 같이 블록 데이타들은 양자화 이전의 DCT 블록과 전환되는 장면의 DCT 블록과 함께 가중치 평균을 구하여 이를 기록하게 된다. 이 DCT 영역에서의 연산은 식(2)에 소개된 원리에 따른 것이다. 블록 데이타 이외의 MPEG 비디오 헤더에 포함되어 있는 정보는 원래의 값에 기초하여 전환 효과가 처리된 I 프레임이 최종적으로 생성되게 된다. 이때 양자화 단계값은 원본의 것을 그대로 사용하게 된다. 그리고, P 프레임의 움직임 보상 처리를 위하여 원시 형태까지 디코딩을 수행하여 <그림 5>에서 보이는 프레임 버퍼에 저장하게 된다.

원시 데이타로 완전 복원하는 방법과의 차이점은 I 프레임의 원시 형태로의 복원이 단지 P 프레임의 생성을 위한 것일 뿐이라는 것이고, I 프레임 그 자체의 생성은 DCT 영역에서 이루어진다는 점이다. 또한 기존의 DCT 영역에서의 효과 처리 방법과 달리 양자화 과정을 거치지 않는다. 이로써 생성되는 I 프레임은 화질의 저하를 초래하지 않아도 될 뿐 아니라, DCT를 거치지 않을 수 있기 때문에 빠른 수행 시간을 얻을 수 있게 된다. I 프레임은 GOP(Group of picture)내의 화질을 좌우한다는 사실을 생각해 볼 때에 이는 매우 큰 장점이라 할 수 있다.

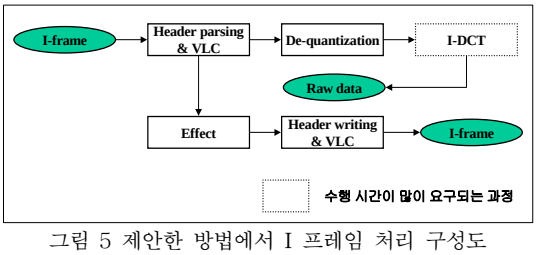


그림 5 제안한 방법에서 I 프레임 처리 구성도

3.1.2 B 프레임의 처리

B 프레임의 생성되는 과정은 <그림 6>을 통해서 알 수 있듯이 움직임 보상도 하지 않고, 움직임 예측도 하지 않으며, IDCT 및 DCT과정도 거치지 않는다. 이는 입력되는 프레임 내에 기록된 움직임 벡터를 그대로 사용하기 때문이며, 두 프레임의 겹쳐지는 비율이 정해져 있기 때문에 이를 산술적으로 계산할 수 있고, 이를 블록에 기록해 줄 수 있으며, 이 과정에서 필요한 움직임 보상은 근사처리 방법을 사용하고 있기 때문이다.

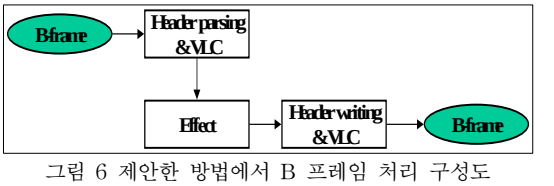
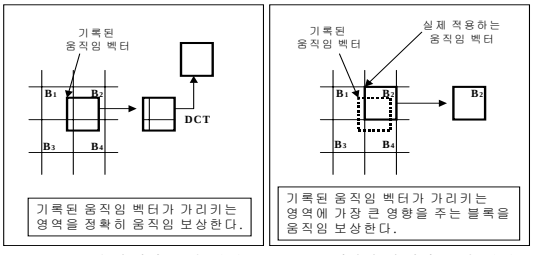


그림 6 제안한 방법에서 B 프레임 처리 구성도

먼저, 기존의 움직임 보상 방법과 제안한 근사처리 방법에 대하여 설명하겠다. 움직임 보상은 화소단위 또는 반화소 단위의 16×16 크기의 영역에 대하여 이루어지기 때문에 앵커 프레임의 원시 형태로의 복원이 반드시 요구된다. 그리고, 이들 영역 내 6개의 8 x 8크기 블록값들을 다시 DCT한 후, 현재 프레임의 DCT 영역에서의 블록값과의 차이를 구하게 된다. 이는 B 프레임내 대부분의 매크로블록이 앵커 프레임을 참조하는 특성을 생각해 볼 때, 전체 프레임에 대하여 DCT를 수행하는 계산량이 필요하게 된다.

하지만, 제안한 움직임 보상법은 <그림 7>에서 보는



(a) 기존의 움직임 보상 방법 (b) 제안한 움직임 보장 방법

그림 7 움직임 보상 방법의 비교

바와 같이 움직임 벡터가 가리키는 영역이 가장 많이 포함되는 DCT 영역 블록을 선택하여 참조하게 되므로, 비록 근사값을 구하게 되지만 빠른 처리가 가능하게 된다. 만일 이 움직임 보상법을 MPEG 비디오 상영기에서 사용한다면, 화질이 매우 떨어지는 화면을 보게 될 것이다. 하지만, 이 방법은 두 그림의 가중치 비율에 의한 합성에 의해 변화되는 부분을 추정하기 위해 쓰이는 것이므로, 그렇게 정확한 값을 구할 필요가 없다.

B 프레임은 시간적으로 I 프레임과 P 프레임의 사이, 또는 P 프레임 사이에 위치하게 되지만, 실제 저장되는 순서는 참조하여야할 두 앵커 프레임 뒤가 되기 때문에, 전 단계의 DCT 블록 값들을 간직하고 있다가, 이 블록값들에 대하여 움직임 보상 근사 처리를 해 주게 된다. 따라서, 이미 DCT된 블록값들을 이용할 수 있기 때문에 본 논문에서 제안하는 움직임 보상법은 단지 참조하는 블록값과 현재 B 프레임 내 블록값 사이에서 덧셈 연산만을 수행하게 될 뿐이다. 이렇게 움직임 보상을 통하여 현재 처리 중 인 B 프레임 값의 근사값을 알아낸 후에는 합성해야할 그림과의 가중치 평균을 구하고, 이와의 차이를 구하여 이 값을 B 프레임 내 블록에 기록해야 한다. 만일 B 프레임 내 매크로블록 중 움직임 벡터에 의해 참조만 하고, 예측 에러를 기록하지 않는 경우가 있다면, 블록이 존재할 수 있도록 매크로블록 형식을 바꾸어 주어야 한다. 이 블록에 기록될 내용(X)은 식(3)에 의해 표현되는 값이 되는 데, 여기서 A와 B는 입력 스트림의 프레임을 나타내고 T는 효과가 부여된 프레임을 나타낸다. 특히 페이드의 경우 B는 검정색 프레임을 의미한다.

$$\alpha A_1 + (1-\alpha)B_1 = T_1$$
$$\beta A_2' + (1-\beta)B_2 = T_2$$
$$A_2' = A_1 + A_2$$
$$X = T_2 - T_1$$
$$X = \beta\left[\frac{\{T_1 - (1-\alpha)B_1\}}{\alpha} + A_2\right] + (1-\beta)B_2 - T_1 \qquad (3)$$

이 식을 좀 더 자세히 설명하면 다음과 같다. <그림 8>에서 보는 바와 같이 움직임 보상을 위하여 참조하게 되는 앵커 프레임은 이미 가중치 평균에 의해 합성된 내용으로 저장되어 있다. 따라서, 순방향 또는 역방향으로 참조를 해야하는 경우에 합성되기 전의 본래의 값을 구해야 하고, 이 값을 토대로 움직임 보상 근사 처리를 한 후, 현재 프레임에 해당하는 가중치 평균값과 참조되는 프레임과의 차이를 기록해야 한다. 따라서, 합성된 그림으로부터 원래의 앵커 프레임의 내용을 계산해 내어야 한다. 이는 간단한 연립 방정식을 풀게 되면 해결되며, 이로부터 유도된 식이 바로 식(3)이다.

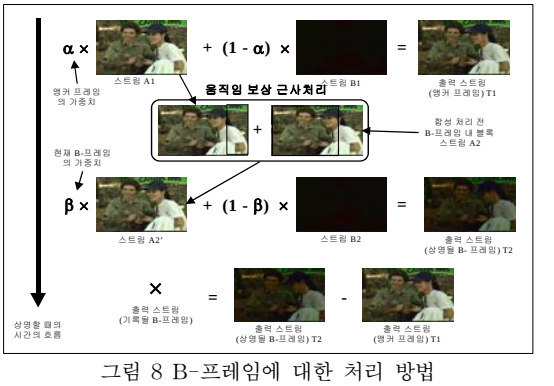


그림 8 B-프레임에 대한 처리 방법

연립 방정식 중 첫 번째 식은 <그림 8>의 맨 윗 그림에 해당하는 것으로, T1이 DCT 영역에서 스트림 A_1과 스트림 B_1의 가중치 평균값으로 구해지는 것을 나타낸다. 둘째 식은 현재 합성되지 않은 B 프레임의 reconstructed[5]된 화면에 대하여 가중치 평균을 구하는 값으로, 실제 reconstructed되는 과정은 움직임 보상 근사 처리 방법을 사용하게 된다. 세 번째 식은 근사 처리에 의한 움직임 보상에 의해 생성되는 reconstructed된 화면을 구하는 것으로, 움직임 벡터만 기록된 매크로블록의 경우에는 $A_2' = A_1$이 성립된다. 네 번째 식은 최종적으로 생성될 B 프레임의 블록에 기록될 값을 나타낸다. 따라서, 마지막에 나타난 식은 앞의 4개의 식을 이용하여 미지수를 소거한 것을 정리한 것이다. 이 식과 <그림 8>은 편의상 프레임 단위로 설명하고 있지만, 실제로는 매크로블록 단위로 적용이 된다. 또한 양방향 참조가 일어나는 경우에는 순방향 참조에 의한 X값과 역방향 참조에 의한 X값의 평균값을 구하여 기록하게 된다.

3.1.3 P 프레임의 처리

P 프레임을 디코딩하는 과정은 완전히 디코딩하는 기존의 방법과 동일하다. <그림 9>에서 보는 바와 같이

P 프레임을 생성하는 과정에서 움직임 벡터를 찾아내는 과정이 없다. 이는 모든 프레임 내 매크로블록을 인트라 형식으로 코딩하기 때문이다. 이 과정을 자세히 설명하자면 다음과 같다. <그림 5>의 프레임 버퍼에 저장된 원시 데이타는 <그림 9>의 것과 동일하다. 이 원시 형태로 복원된 앵커 프레임 데이타를 이용하여 움직임 보상을 거쳐 reconstructed picture를 구성하고, 이산 코사인 연산을 통하여 변환한 후에 이들 블록을 인트라 형식으로 구성한다. 이때, 양자화 단계는 가장 낮은 단계를 선택하여, 좋은 화질을 얻을 수 있도록 한다. 프레임의 형식을 비롯한 픽쳐 계층의 다른 정보들은 변화를 주지 않는다. 이 과정은 앞서 설명한 I 프레임이나 B 프레임을 생성하는 과정보다 많은 수행시간을 요구하게 된다. 따라서 본 논문에서 제안하는 시스템의 성능은 입력으로 받아들이는 비디오 스트림내 P 프레임이 차지하는 비율이 적을수록 좋아지게 된다.

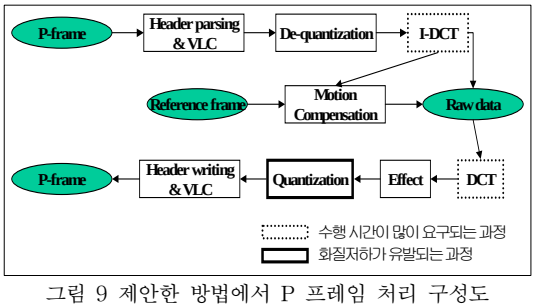


그림 9 제안한 방법에서 P 프레임 처리 구성도

한편, 앞에서 설명한 B 프레임 처리 방법은 인트라 형식으로 코딩된 앵커 프레임을 참조할 수 있도록 설계된 MPEG의 특성을 살린 것으로, 비슷한 구조를 가진 P 프레임에 대하여도 적용할 수 있을 것이다. 즉, 이 방법을 P 프레임에 대해서 적용한다면, 전체적인 수행시간은 더욱 짧아 질 것이다. 하지만, <그림 10>에서 보여지듯이 P 프레임은 다른 P 프레임 및 B 프레임들이 계속적으로 참조를 하기 때문에, 해당 GOP내의 처음의 작은 계산 오차가 시간적으로 뒷부분에 위치한 B 프레임에서는 심각한 오차로 확대된다. 반면 <그림 11>에서처럼 B 프레임은 비록 작은 에러가 있더라도 해당 프레임에서 종료가 될 뿐 에러의 전파가 이루어지지 않는다. 따라서, GOP의 길이가 짧은 스트림의 경우에는 B 프레임 처리 방법을 P 프레임에 적용하여도 좋은 결과를 얻을 수 있을 것이다.

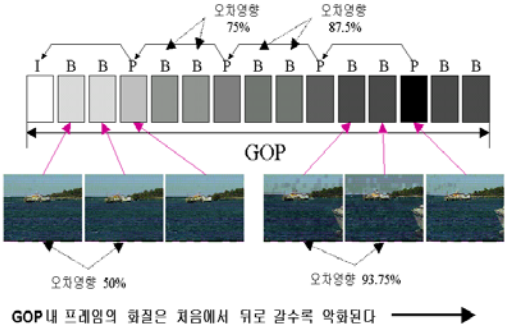


그림 10 P-프레임에 대하여 움직임 보상 근사처리 방법을 적용하였을 경우의 오차 영향

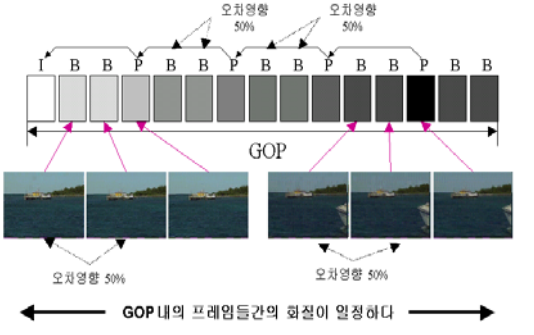


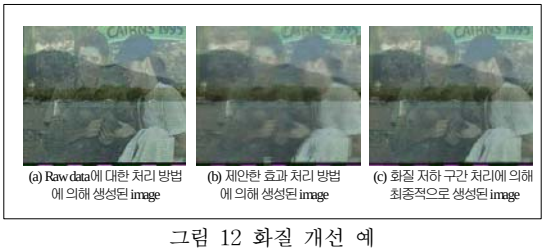
그림 11 P-프레임에 대하여 움직임 보상 근사처리 방법을 적용하지 않을 경우의 오차 영향

3.2 페이드 인 및 페이드 아웃

페이드 인 및 페이드 아웃 효과를 앞서 설명한 기본 처리 방법을 이용하여 처리하려면, 입력으로 받아 들이는 복수 개의 비디오 스트림 대신에 하나의 단위 블록과 하나의 비디오 스트림으로 간소화하면 된다. 즉, 미리 DCT 영역까지 인코딩된 검정색 단위 블록 하나와 정상적인 MPEG 비디오 스트림에 대한 적용이라 할 수 있다. 검정색 단위 블록은 어떤 양자화 단계값을 가지더라도 별다른 영향이 없기에 DCT 영역에서 인트라 블록간 가중치 평균이 이루어질 때에 발생되는 양자화 단계값의 차이에 의한 블록화 현상(blockiness)[15]이 일어나지 않는다. 그리고, 움직임 벡터를 사용하지 않으므로, 다른 입력으로 들어오는 비디오 스트림의 움직임 벡터를 그대로 사용하더라도 아무런 문제가 발생하지 않는다. 검정색 DCT 블록은 루미넌스(Luminance)블록의 경우 DC value = -900, AC value = 0으로 주면 되고, 크로미넌스(Chrominance) 블록의 경우 모든 값을 0으로 하면 된다.

3.3 디절브

일반적으로 MPEG 인코더들은 시간적으로 순방향 또는 역방향으로 움직임이 빈번하게 많은 경우에는 인트라 형식의 매크로블록을 사용하거나, 주어진 비트량만으로 해당 프레임내의 영상을 표현하기 위하여 양자화 단계를 조절하여, 화질을 떨어뜨리게 된다. 프레임간 코딩 형식으로 저장된 일반적인 MPEG 비디오 스트림들에 대하여 제안한 방법, 즉 움직임 예측 과정을 생략하면서 이 디절브 효과를 처리하기 위해서 가장 문제가 되는 것은 두 비디오 스트림 중에서 어떤 스트림의 움직임 벡터를 사용할 것인지에 대한 것이다. 만일 현재 장면을 간직하고 있는 비디오 스트림의 움직임 벡터를 사용한다면 전환의 마지막 부분에서의 움직임 벡터가 여전히 처음 비디오 스트림의 내용에 기초하여 생성된 것이기 때문에 전혀 엉뚱한 영역을 가리키고 있으므로, 화면이 마치 각 블록이 임의의 방향으로 조금씩 밀린 모자이크 그림과 같이 된다. 일반적으로, 화면의 움직임이 빈번한 경우에는 화면 내에 인트라 형식의 매크로블록의 개수가 많아지는 특성이 있고, 화면의 움직임이 많지 않은 경우에는 움직임 벡터에 의한 프레임간 참조가 많이 일어나며, 이때의 움직임 벡터의 크기는 그리 크지 않다. 이러한 특성을 기저에 두고서, 전술한 기본 처리 방법에 몇 가지 부가적인 과정을 추가하여 디절브 효과의 처리 방법을 설명하고자 한다. 이러한 추가적인 루틴에 의해 개선된 예는 <그림 12>에 나타내었다.



(a) Raw data에 대한 처리 방법에 의해 생성된 image (b) 제안한 효과 처리 방법에 의해 생성된 image (c) 화질 저하 구간 처리에 의해 최종적으로 생성된 image

그림 12 화질 개선 예

앞서 설명한 기본 처리 과정을 보면, 비디오 데이타뿐만 아니라 각종 헤더 정보를 제공하게 되는 주가 되는 스트림과 단지 DCT 영역의 값만을 제공하는 보조가 되는 스트림으로 나누어서 생각해 볼 수 있다. 페이드 인 및 페이드 아웃의 경우, 비디오 스트림이 하나밖에 없으므로, 이에 대한 고려가 없었지만, 디절브의 경우, 두 개의 비디오 스트림이 그 대상이 되므로, 이런 분류를 해야 한다. 따라서, 가중치가 0.5보다 크게 적용 받는 스트림을 주가 되는 스트림이라 하고, 그렇지 못한 경우를 보조가 되는 스트림이라 정하도록 한다. 보조가 되는 스트림은 DCT 영역의 블록값 만을 제공하게 된다.

(1) 양자화 단계값의 결정

먼저 생각해 보아야 할 점은 두 블록간의 양자화 단계값의 차이에 대한 조정이다. 본 논문에서 제안한 방법에서는 이러한 고려를 하지 않는 데, 이는 가중치를 많이 적용 받는 쪽의 양자화 값을 살리는 것이 오히려 효과가 더 좋기 때문이다. 하지만, 이 방법은 진행률이 50% 정도 되었을 경우, 블록화 현상이 발생하는 단점이 있다. 이러한 중간 구간에서 I 프레임에 대하여는 양자화 단계값을 두 스트림의 가중치 평균값을 사용하도록 하였다. 이는 비록 근사치이기는 하지만, 영상의 8 x 8 크기의 블록화 현상을 방지하는 역할을 하게 된다. P 프레임의 경우 기본 처리 과정에서 언급하였듯이 양자화 단계값이 동일하기 때문에 아무런 문제가 발생하지 않는다.

(2) 움직임 벡터 설정 방법

다음으로 고려해야 할 사항은 움직임 벡터의 사용 문제이다. 기본적으로 움직임 벡터는 주가 되는 스트림내의 B 프레임의 것을 사용하게 된다. 높은 가중치를 받는 프레임의 특성을 그대로 살리기 위함이다. 하지만, 중간 구간이나, 두 스트림의 B 프레임내 같은 위치의 매크로블록에 기록된 움직임 벡터의 차이가 심하게 나는 경우에는 화면이 일그러지는 현상이 발생하게 된다. 따라서, 이러한 문제는 부가적인 과정을 추가하여 해결해야만 한다. 이를 해결하기 위한 방안은 <그림 13>에서 보는 바와 같이, 시간적으로 과거와 미래에 해당하는 두 앵커 프레임을 양방향으로 참조만 하도록 매크로블록 형식을 바꾸는 것이다. 이때의 움직임 벡터의 값은 0으로 하게 된다. 이는 두 앵커 프레임을 0.5의 비율로 디절브한 결과를 얻게 된다. 일반적인 MPEG 비디오 스트림의 경우 하나의 GOP안에 15개의 프레임이 있게 되고, 이 중 하나의 I 프레임과 4개의 P 프레임, 10개의 B 프레임이 있게 된다. 이런 경우, 두 앵커 프레임 사이의 B 프레임의 개수는 2개가 되며, 이들은 두 앵커 프레임의 중간 단계의 모습을 표현하게 된다. 따라서, 이보다 적은 수의 B 프레임을 가지는 경우에는 화질이 더욱 좋아질 것이며, 그 반대의 경우에는 화질이 좀 더 떨어질 것이다.

(3) 중간 구간의 결정 방법

앞서 언급한 양자화 단계값 및 움직임 벡터 처리 방법은 가중치 비율의 차이가 별로 없는 중간 구간에서 이루어진다. 이때, 임계값을 어느 정도로 설정해 주고, 이 중간 구간을 미리 결정해 주어야 한다. 먼저, 움직임 벡터 처리 방법에 대한 중간 구간 결정을 위한 임계값

은 두 스트림의 B 프레임 상에서 같은 위치에 있는 매크로블록의 움직임 벡터값의 차이의 합을 구한 후, 보조가 되는 스트림의 가중치 비율을 곱하여 얻는다. 이 방법을 적용하게 되면, 처음과 끝은 언제나 임계값이 0이 되며, 같은 조건일 경우 전체 효과 처리 구간 중 정 가운데에 위치한 프레임의 임계값이 가장 커지게 된다. 다음으로 양자화 단계값의 처리를 위한 중간 구간은 가중치 비율이 0.25에서 0.75 사이일 때로 정하였다. 일반적으로 얻을 수 있는 영화나 드라마의 비디오 데이타를 예로 들어 볼 경우, 움직임이 적당한 부분에 대하여 처음 30%정도부터 나중 70%정도까지를 중간 구간에 대한 임계값을 측정하여 적용하게 되면 좋은 화질을 얻을 수 있다.

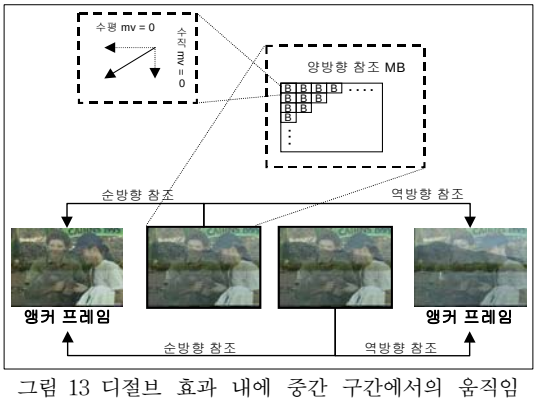


그림 13 디절브 효과 내에 중간 구간에서의 움직임 벡터 처리

4. 성능 평가 및 분석

본 논문에서 제안한 압축 영역에서의 전환 효과 처리 방법을 실제 구현을 통하여 효과 처리 결과 화면의 화질과 생성되는 시간을 측정하여 그 성능을 평가하였다. 또한 이런 성능을 일반적인 방법(원시 형태의 데이타에서의 처리 방법), I 프레임으로 변환하여 처리하는 방법, 제안한 방법에 대하여 비교하여 제안한 방법의 화질 유지 능력과 생성 속도가 기존의 방법보다 더 나은 성능이 향상되었다는 것을 보이겠다. 효과기는 일반적인 형태의 MPEG 1 비디오 스트림을 받아들여 효과를 가한 후에 다시 MPEG 1 비디오 스트림을 생성하도록 만들어 졌으며, 비교 대상이 되는 두 효과기 역시, 효과를 가하는 부분을 제외한 모든 과정, 즉 DCT 및 그 역 변환, 움직임 보상 및 예측, 양자화 및 그 역과정과 VLC 및 그 역과정을 비롯한 모든 부분은 동일한 루틴[4, 5]으로 이루어졌다. 또한, 화질의 비교 분석은 PSNR(Peak Signal-to-Noise Ratio)값을 통하여 비교하였으며, 이를 위하여 MPEG 비디오 스트림을 원시 형태의 데이타로 복원해 주는 디코더[4]와 원시 형태의 데이타를 프레임 별로 입력받아서 다음과 같은 PSNR 값[9]을 구해주는 루틴을 추가적으로 구현하였다.

$$PSNR = 10\log_{10}\frac{255^2}{\frac{1}{MN}\sum_{i=1}^{M}\sum_{j=1}^{N}(X_{ij}-Y_{ij})^2} \quad dB \qquad (4)$$

윗 식에서 M, N은 비교하는 그림의 가로 및 세로의 길이이고, X 및 Y는 비교하는 두 그림의 좌표 i, j에서의 화소값이다. 성능 측정에 사용되는 비디오 스트림은 정지된 화면, 움직이는 화면 줌-인(Zoom-in)과 같은 카메라 작용이 포함된 특성을 가지고 있으며 길이는 3초이다. GOP의 구조는 I 프레임 1개, P 프레임 4개, B 프레임 10개로 구성된 형태이다.

4.1 실험 결과 분석

먼저, 3개 방법에 대하여 프레임 형식 별 생성 속도를 비교하여, 본 논문에서 제안한 방법을 통해서 이득을 얻는 구체적인 단계를 규명하고, 그 후 페이드 인 및 페이드 아웃과 디절브에 대하여 화질 및 생성 속도의 비교를 통한 성능 분석을 할 것이다. 마지막으로, 스트림의 특성 별 화질의 비교를 통하여 본 처리 방법의 장단점을 분석함으로써, 전체적인 성능 평가를 마칠 것이다.

4.1.1 프레임 생성 시간

본 실험은 각 효과 처리 방법에 있어서 프레임 형식별로 생성하는 데에 소요되는 시간을 측정함으로써, 어떤 단계에서 제안한 방법이 효율성을 나타내는지를 알아보기 위하여 시도하였다. 전환 효과 중에서 페이드 인을 선택하여 3가지 방법에 의해 각 프레임 형식 별로 생성되는 시간을 측정하였으며, 다른 효과 처리 방법의 경우에는 페이드 인과 비슷한 결과를 얻을 수 있으므로, 이에 대한 결과는 생략하였다. 그 결과는 <표 1>로 나타내었으며, <그림 14>는 이 표를 알아보기 쉽게 나타낸 것이다. 결과를 보면, 프레임 형식 별로 생성 시간에 있어서 차이를 보이고 있는 데, 여기에 영향을 준 요소는 DCT와 움직임 보상 및 예측 과정이라 볼 수 있다. 기타 다른 과정이 프레임 생성 시간에 미친 영향은 시간으로 측정하기 힘들 정도로 작다. 제안한 방법에서의 I 프레임의 생성 시간이 원시 형태 데이타 및 I 프레임만으로 이루어진 스트림에서의 처리 방법에서보다 적게 나오는 이유는 인코딩하는 단계에서 DCT 연산을 수행하지 않기 때문이다. I 프레임 형태에의 방법이 원시 형태에서의 방법보다 더 적게 나오는 이유는 앵커 프레임으로서 프레임 버퍼로 복사되는 오버헤드가 없기 때문

이다.

표 1 프레임 별 생성 시간의 비교

프레임 진행	제안한 방법	I frame only stream에 대한 처리	원시 데이타에서의 처리
I - Frame	0.50 초	1.48 초	1.54 초
B - Frame	0.38 초	1.54 초	5.05 초
B - Frame	0.49 초	1.53 초	5.11 초
P - Frame	1.32 초	1.49 초	3.79 초
B - Frame	0.44 초	1.54 초	5.11 초
B - Frame	0.50 초	1.54 초	5.16 초
P - Frame	1.37 초	1.49 초	3.68 초
B - Frame	0.38 초	1.53 초	5.11 초
B - Frame	0.50 초	1.53 초	5.11 초

다음으로, P 프레임의 경우를 살펴보자. 3가지 방법 모두 디코딩하는 과정은 동일하다고 볼 수 있다. 하지만, 제안한 방법과 I 프레임만으로 이루어진 스트림에 대한 방법의 경우 움직임 예측을 하지 않는 반면, 원시 형태에 대한 방법에서는 이에 대한 계산 부담을 안고 있어서 좀 더 긴 수행 시간을 필요로 한다.

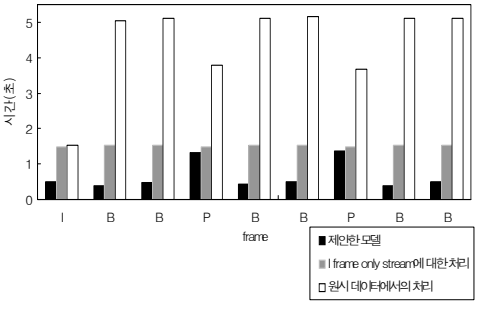


그림 14 프레임 형식 별 생성 시간

마지막으로, B 프레임의 경우에 대하여 분석하여 보겠다. 제안한 방법에서는 DCT 및 그 역과정을 수행하지 않는다. 그리고 움직임 예측과정이 생략되며, 다만, 제안한 방법에 의한 움직임 보상과 변화된 정보를 추가적으로 기록하는 정도의 오버헤드만을 안고 있을 뿐이다. 반면 원시 형태에서의 방법의 경우, 전술한 P 프레임의 계산량에 추가되는 역방향에 대한 움직임 보상 및 예측에 대한 부담도 안고 있으므로, 프레임을 생성하는 데에 가장 많은 시간을 소비한 것으로 나타났다. I 프레임만으로 이루어진 스트림에 대한 방법에서는 프레임 형식간의 차이가 거의 나타나지 않았으며, 다만 B 프레임의 생성시간이 조금 더 길게 측정되었을 뿐이다. 이는 양방향으로 움직임 보상을 하기 때문에 P 프레임의 경우보다 시간이 더 걸린 것이라 볼 수 있다. 결과적으로 본 처리 방법은 기존의 방법에 비하여 생성 속도 측면에서 전체적으로 나은 성능을 보이고 있으며, 특히 가장 생성 속도가 느린 B 프레임에 대하여 개선 효과가 높은 것으로 나타났다.

표 2 각 적용 방법 별 처리 속도의 비교

적용방법 / 전환효과	제안한 방법	I frame only stream에 대한 처리	원시 데이타에서의 처리
페이드 인	1.36 프레임/초	0.65 프레임/초	0.30 프레임/초
페이드 아웃	1.37 프레임/초	0.65 프레임/초	0.31 프레임/초
디절브	0.99 프레임/초	0.47 프레임/초	0.28 프레임/초

스트림의 생성 속도의 비교는 <표 2> 및 <그림 15>를 통하여 나타내었다. 제안한 방법이 I 프레임으로 이루어진 스트림에 대한 방법보다는 약 2배, 원시 형태에 대하여 처리하는 방법에 비해서는 3배 이상의 성능을 보이고 있다.

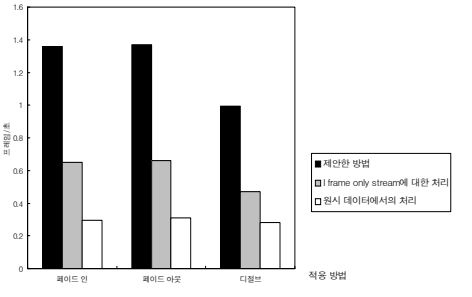


그림 15 각 적용 방법 별 처리 속도의 비교

4.1.2 화질

(1) 처리 방법 별 화질 비교

<그림 16>, <그림 17>, <그림 18>은 페이드 인 및 페이드 아웃과 디절브 효과에 대하여 화질 측면에서의 비교 실험 결과를 나타낸 것이다. 그래프의 구간은 편집 효과의 시작 시점부터 끝나는 시점까지이며, 실험 결과로서 나타낸 3개의 그래프는 프레임별 PSNR값으로, R, G, B형태의 원시 형태의 데이타에 대하여 각 색깔별로

PSNR값을 구한 후 3개의 색에 대하여 평균을 구하여 나타내었다. 기준이 되는 프레임은 원래의 스트림을 별도의 디코더를 이용하여 복원하여 원시 형태로 만든 후에 각 R, G, B값에 대하여 검정색 또는 대상 프레임에 대하여 overlapping을 시도하였다. 기준과 비교하여야 하는 두 개의 스트림은 각 효과기가 생성해낸 MPEG 비디오 스트림을 위의 디코더를 이용하여 원시 형태로 복원한 후에 PSNR값을 구하였다.

페이드 인 및 페이드 아웃의 경우, 본 논문에서 제안한 방법이 전체 프레임 구간에 걸쳐서 좋은 화질을 얻을 수 있었다. 이는 검정색 단위 블록과의 합성이 양자화 및 움직임 예측 및 보상에 별 영향을 미치지 않은

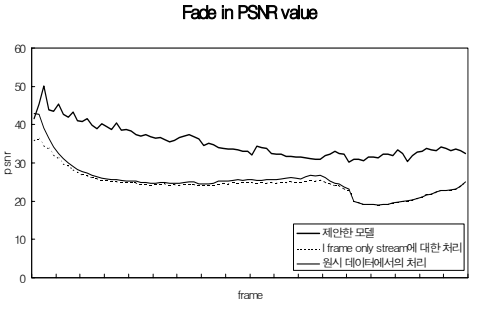


그림 16 페이드 인 효과에서의 화질 비교

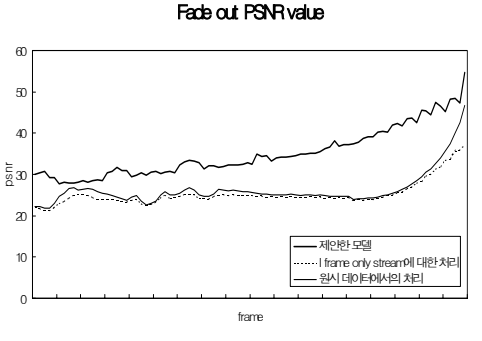
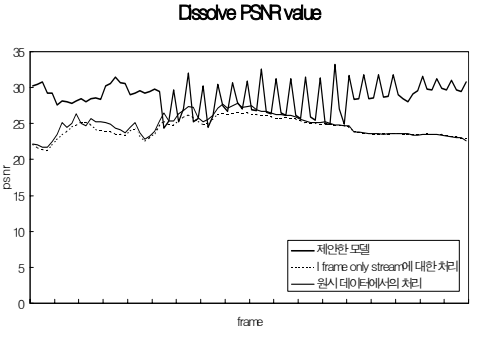


그림 17 페이드 아웃 효과에서의 화질 비교

그림 18 디절브 효과에서의 화질 비교

결과라 볼 수 있다. 제안한 방법에 의해 생성된 프레임들의 화질은 입력으로 받아들이는 I 프레임의 화질에 영향을 거의 주지 않은 채로 전환 효과를 가할 수 있는 특성에 의해 잘 유지되었다. 모든 프레임을 원시 형태로



복원한 후 이를 다시 인코딩하는 나머지 두 방법에 의해 생성되는 프레임의 경우도 화질 유지 능력에 있어서 비슷한 결과를 보여주고 있다.

<그림 18>는 디절브 효과에 의한 화질 비교실험의 결과를 나타내고 있다. 이 결과를 보면, 3가지 방법에 대하여 거의 비슷한 화질을 보여주고 있다. 다만 제안한 방법에 의해 생성된 프레임들의 경우 중간 구간에서 화질이 고르지 못한 면을 보여주는 데, 이는 3장에서 설명한 양자화 단계값의 및 움직임 벡터의 상이성에 의한 것이다. 특히 중간 구간에서 화질이 떨어지는 부분은 B 프레임에 해당되는 데, 영상이 깨지는 현상을 막기 위해 프레임 내 모든 매크로블록이 양방향 참조를 하도록 인위적으로 움직임 벡터를 조정하였기 때문이다.

이와 같은 실험을 통하여 제안한 방법에 의해 생성해낸 비디오 스트림의 화질은 3가지 전환 효과에 대하여 모두 원래 화질을 유지할 수 있음을 알 수 있었다.

(2) 스트림 특성별 화질 분석

본 논문에서 제안한 디절브 효과의 처리 방법에 의한 화질은 입력되는 비디오 스트림의 내용에 따라 영향을 받는 특성을 가지고 있다. <그림 18>에 나타낸 실험에 사용된 스트림들은 움직임이 어느 정도 있고, 줌 인(Zoom in) 카메라 작용이 있는 가하면, 움직임이 거의 없는 구간도 존재하는 등의 평범한 특성을 지니는 것들이다. 움직임이 어느 정도 있는 구간에서 움직임 벡터의 사용이 많아지기 때문에 프레임 형식에 따라 화질이 고르지 못한 특성을 보여주었다. <그림 19>에 나타낸 실험 결과는 프레임 내 객체들의 움직임이 별로 없는 특성을 지니는 스트림간의 전환 효과에 의한 화질의 비교이다. 앞서 실험 결과보다 프레임간의 화질의 차이가 많이 줄어들었음을 확인할 수 있다. 따라서, 본 논문에서 제안한 전환 효과 처리 방법 중에서 페이드 인 및 페이드 아웃 효과는 스트림의 내용에 관계없이 기존의 처리 방법과 비슷한 화질 유지 능력을 보이며, 디절브 효과의

경우에는 움직임이 별로 없는 스트림에서 고른 화질을 유지할 수 있음을 알 수 있었다.

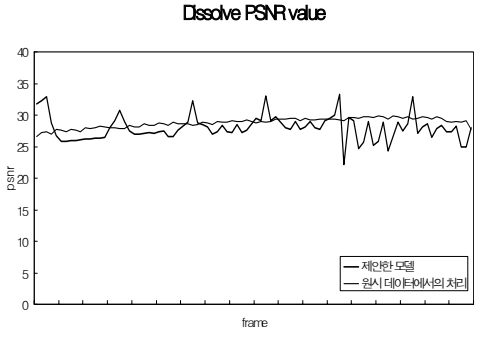


그림 19 움직임이 적은 스트림간 디절브 효과에서의 화질 비교

5. 관련 연구와의 비교

실험한 결과를 토대로 관련 연구와의 비교를 <표 3>에 나타내었다. 이는 fade in/out 및 dissolve효과에 대한 화질, 수행시간, 처리대상 스트림의 특성에 따르는 성능의 영향과 출력되는 비디오 스트림의 용량에 대한 성능 평가에 대한 종합적인 정리이다. <표 3>의 각 항목별로 비교 실험에 대한 결론을 내려보자면 다음과 같다. 먼저, 화질은 본 논문에서 제안한 방법이 다른 2가지 방법에 의한 실험에서보다 다소 좋은 결과를 얻었지만, 일부 실험에서는 좋지 못한 경우도 있었기 때문에 전체적으로 3가지 실험 방법에 대하여 비슷한 화질을 얻었다고 결론 내릴 수 있다. 다음으로 수행시간은 제안한 방법이 기존의 방법보다 2배에서 4배 정도 향상된 결과를 얻을 수 있었다. 다음으로 스트림의 내용 특성에 의한 성능 영향은 제안한 방법에 의한 디절브 효과의 적용시 개체의 움직임이 커짐에 따라 화질이 저하되는 문제점이 발견되어, 이에 의한 영향이 없는 다른 2가지 방법에 비해 단점으로 지적될 수 있다. 다음으로 스트림 내 프레임 구성에 따른 성능의 영향에 있어서는 제안한 시스템의 특성상 B 프레임의 구성비가 많아질수록 처리 속도도 크게 향상되는 반면, I frame only stream에 대한 처리 방법에 대해서는 I 프레임이 많을수록 처리 속도가 향상된다. 일반적인 MPEG 비디오 스트림의 경우 B 프레임이 2/3이상을 차지하므로, 이러한 특성은 제약사항이라 할 수 없다고 생각된다. I frame only stream에 대한 처리 방법의 경우 I 프레임은 스트림내 구성비가 매우 적기 때문에 단점이라 할 수 있으며, Raw data에 대한 처리 방법은 스트림내 프레임의 구성과 상관없이 일정한 성능을 보여준다. 마지막으로 출력 스트림의 용량에 대한 것은, 입력받는 스트림의 용량과 비교했을 때에 제안한 방법은 P 프레임의 인트라 코딩에 대한 증가분, I frame only stream에 대한 처리 방법에서는 B, P 프레임의 인트라 코딩에 대한 증가분이 추가된 결과를 얻을 수 있었다.

표 3 관련 연구와의 비교 (Fade in/out, Dissolve)

비교 대상 \ 적용방법	제안한 방법	I frame only stream 에 대한 처리[1]	Raw data에서의 처리 [12]
화 질	25 - 45dB	20 - 40dB	20 - 40dB
수 행 시 간	1.24 frame/s	0.59 frame/s	0.30 frame/s
스트림의 내용 특성에 따른 성능 영향	개체의 움직임이 적을수록 좋은 성능	일정	일정
스트림 내 프레임 구성에 따른 성능 영향	B 프레임이 많을수록 좋은 성능	I 프레임이 많을수록 좋은 성능	일정
출력 스트림의 용량	원본보다 약간 증가	원본에 비해 현저히 증가	원본과 같음

6. 결론 및 앞으로의 연구 방향

MPEG 비디오에 대한 편집 작업에서 전환 효과는 서로 다른 내용을 가지는 장면들을 편집하는 과정에서 장면 전환이 이루어지는 경계에 전환이 부드럽게 이루어지게 하기 위하여 많이 사용된다. 소프트웨어 형태의 편집기 시스템은 많이 개발되어 있으나, 효과를 가한 후에 화질이 저하되는 문제점과 함께 지나치게 작업시간이 오래 걸리는 단점을 안고 있었다. 본 논문에서는 많은 전환 효과 중에서 가장 널리 사용되는 페이드 인 및 페이드 아웃 효과와 디절브 효과를 처리하는 새로운 방법에 대하여 제안하고 구현하였으며, 제안한 방법은 화질 저하를 막을 수 있도록 양자화 단계를 최소화하였으며, 수행시간의 단축을 위하여 DCT 과정을 가능하면 사용하지 않고, 움직임 예측 과정을 생략하도록 설계되었다. I 프레임과 P 프레임의 생성 원리는 기존의 연구 방법을 응용하여, 인트라 형식의 매크로블록을 사용하여 인코딩 하도록 하였으며, B 프레임은 근사 처리에 의한 움직임 보상법을 이용하여 처리 중인 스트림 내 움직임 벡터를 그대로 사용할 수 있도록 하여 DCT 과정과 움직임 예측 과정을 거치지 않도록 하였다. 디절브의 경우

일부 구간에서 움직임 벡터의 크기를 0으로 만들고 양방향 참조를 하는 B 프레임을 사용하여 두 스트림의 합성에 의해 발생되는 움직임 벡터의 상이성에 의한 화면의 깨짐 현상을 극복하였다.

본 논문에서는 MPEG 비디오 스트림에 대하여 전환 효과를 가하기 위한 일반적인 방법의 각 단계를 분석하여 효율적이지 못한 과정을 최적화 시킴으로써 3 - 4배 정도의 빠른 수행 시간을 얻을 수 있었다. 기존의 연구 결과에 의한 방법과 일반적인 방법에 의해 생성된 스트림과 비교하였을 때에 비슷한 화질이 유지되는 결과를 얻었다. (실험 결과 그래프에는 제안된 방법에 의해 더 좋은 화질을 얻는 것으로 되어 있는 데, 이는 양자화 - 역양자화 과정을 거치지 않기 때문이며, 샘플이 동일한 인코더에 의해 생성되어서 매크로블록 헤더에 기록되는 mquant값이 비슷한 특성에 의한 현상이다. 따라서, 화질에 있어서는 더 나은 결과를 보인다고 이해할 수는 없다.) 그러나 페이드 인 및 페이드 아웃 효과의 경우 스트림의 내용에 상관없이 고른 화질을 유지할 수 있었으나, 디졸브 효과를 처리하는 데에는 움직임이 많은 스트림에서 화질이 고르지 못한 단점을 보여주었다. 한편, 전환 효과를 처리하는 과정에서 비트율이 고르지 못한 현상이 발생하는 데 이러한 문제점은 앞으로의 연구를 통하여 해결해 나가야 할 것이다.

참 고 문 헌

[1] Chang, S.-F. and Messerschmitt, D.G., "Manipulation and Compositing of MC-DCT Compressed Video," *Proc. of IEEE JSAC Special Issue on Intelligent Signal Processing,* 1994.

[2] Chang, S.-F. and Messerschmitt, D.G., "Video Compositing in the DCT Domain," *Proc. of IEEE Workshop on Visual Signal Processing and Communications,* 1992.

[3] Chang, S.-F. and Messerschmitt, D.G. "A New Approach to Decoding and Compositing Motion Compensated DCT-Based Images," *Proc. of IEEE ICASSP,* 1993.

[4] MPEG Software Simulation Group, MPEG-2 Video Codec, *Available from http://www.mpeg.org/MPEG/MSSG.*

[5] ISO/IEC/JTC1/SC29/WG11, *Coding of Moving Pictures and Associated Audio for digital storage media at up to about 1.5 Mbit/s,* ISO/IEC International Standard 11172-2, 1993.

[6] 대우전자 영상 연구소, *MPEG 비디오*, 연암 출판사, July 1995.

[7] 정제창, *그림으로 보는 최신 MPEG*, 교보문고, 1995.

[8] 정제창, *그림으로 보는 응용 MPEG,* 교보문고, 1997.

[9] Jiying Zhao, Yoshihisa Shimazu, Koji Ohta, Rina Hayasaka, and Yutaka Matsushita, "A JPEG Codec Adaptive to Region Importance," *Proc of ACM Multimedia 96,* 1996.

[10] John F. Koegel Buford, *Multimedia Systems*, ACM Press, 1994.

[11] Simon J. Gibbs, Dionysios C. Tsichritzis, *MULTIMEDIA PROGRAMMING Objects, Environment and Frameworks,* Addison-Wesley Publishing Company, 1995.

[12] Porter, T. and Duff, T., "Compositing Digital Images," *Proc. of SIGGRAPH 84 on Computer Graphics*, 1984.

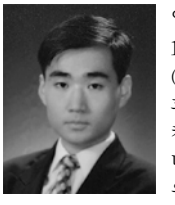
[13] B. Chiptrasert and K. R. Rao, "Discrete Cosine Transform Filtering," *Proc. of Signal Processing*, Vol. 19, No. 3, pp. 233-45, 1990.

[14] Smith, B. C. and Rowe L. "Algorithms for Manipulating Compressed Images," *Proc. of IEEE Computer Graphics and Applications,* 1993.

[15] Albert J. Ahumada, Jr. and Rensheng Horng, "De-blocking DCT Compressed Images," *Proc. of SPIE on Human Vision, Visual Processing, and Digital Display,* Vol. 2179, pp. 109-116, 1994.

[16] B. Shen and I. K. Sethi, "Inner-Block Operation on Compressed Images," *Proc. of ACM Intl. Conf. Multimedia'95*, pp. 490-499, 1995.

[17] B. Shen and I. K. Sethi, "Scanline algorithms in compressed domain," *Proc. of SPIE on Digital Video Compression: Algorithms and Technologies,* Vol. 2668, 1996.



이 승 철

1997년 2월 서강대학교 컴퓨터학과 졸업(학사). 1999년 2월 서강대학교 컴퓨터학과 졸업(석사). 1999년 2월 ~ 현재 청호컴퓨터 (주) 연구원. 관심분야는 멀티미디어 시스템, 이미지 처리, 컴퓨터 네트워크, 인터넷 프로그래밍 및 응용

낭 종 호

1986년 2월 서강대학교 전자계산학과 졸업(학사). 1988년 2월 한국과학기술원 전산과 졸업(석사). 1992년 2월 한국과학기술원 전산과 졸업(박사). 1992년 3월 ~ 1992년 8월 한국과학기술원 정보전자 연구소(연구원). 1992년 9월 ~ 1993년 8월 일본 Fujitsu연구소(방문연구원). 1993년 9월 ~ 현재 서강대학교 전자계산학과 부교수. 관심분야는 멀티미디어 시스템, 병렬 프로그램 언어 및 시스템, 인터넷 프로그래밍 및 응용