논문 2004-41CI-6-10

H.264에서 MPEG-4로 빠른 트랜스코딩

(Fast Transcoding from H.264 to MPEG-4)

권 혁 균*, 이 영 렬**

(Hyouk-Kyun Kwon and Yung-Lyul Lee)

요 약

본 논문은 H.264와 MPEG-4 간의 원활한 통신을 하기위한 두 가지 트랜스코딩 방법을 제안한다. 같은 공간적 시간적 해상도(spatio-temporal resolution)를 유지하는 트랜스코딩 방법과 공간적 해상도(temporal resolution)를 줄이는 트랜스코딩 방법을 제안한다. H.264 비트스트림(bitstream)이 MPEG-4 비트스트림으로 변환 시 H.264 블록형태를 MPEG-4에서 사용 할 수 있는 블록형태로 변환 시켜야 하며, 4×4 블록단위의 움직임 벡터도 8×8 블록단위의 움직임 벡터로 조정하여야 한다. 두 가지 제안된 트랜스코딩 방법은 직렬 화소영역 트랜스코딩 방법(cascade pixel-domain transcoding) 보다 MPEG-4 부호화기 측에서 4.1~5.1배 부호화 속도가 빠를 뿐만 아니라 영상의 화질 저하는 최고 0.3dB정도 밖에 떨어 지지 않는다.

Abstract

This paper proposed two transcodiing methods, which maintain the same spatio-temporal resolution and reduce a spatial resolution, to convert a H.264 video bitstream into an MPEG-4 video bitstream. When the H.264 video bitstream is transformed into the MPEG-4 video bitstream, the conversions between H.264 block types and MPEG-4 block types are performed by minimizing distortion and the 4×4 block-based motion vector mapping is performed. The proposed two transcoding methods run 4.1~5.1 times as fast as the cascaded transcoding methods in MPEG-4 encoder side, while the PSNR (peak-signal-to ratio) is slightly degrade with maximum 0.3dB.

Keywords: 움직임 예측(Motion Estimation), 움직임 보상(Motion Compensation), 매크로블록(Macroblock)

I. 서 론

멀티미디어 통신은 통신 산업에서 가장 빠르게 성장하는 분야 중의 하나이다. 비디오 통신, 디지털 라이브러리(Digital Library), 그리고 VOD(Video on Demand)와 같은 멀티미디어 서비스들이 현재 널리 사용 되어지고 있다. 멀티미디어 응용분야 에서는 여러 가지 통신 채널의 대역폭에 따른 다양한 비디오 비트스트림(video bitstream)에 대한 비트 율(bit rate)이 필요하다. 그래서 송신 측에서 보내어지는 압축 비트스트림 형식을 수신 측 환경에 맞는 비트스트림 형식으로 변환하면서 양측의 QoS (Quality Of Service)를 고려하는 다양한 비디오 트랜스코딩 기법이 발전해 왔다.[1, 2] 일반적으로 비디오 트랜스코딩 기법은 크게 화소영역(pixel-domain)에서의 트랜스코딩 기법[5,6]과 주파수 영역(frequency domain)에서의 트랜스코딩 기법[7,8]으로 분류된다.

본 논문에서는 최근에 적은양의 비트로 고화질의 영상을 제공하는 새로운 동영상 압축표준인 H.264 (ITU-T Recommendation H.264이며 ISO/IEC MPEG-4 Part 10 Advanced Video Coding)[9] Baseline profile (BP)과 기존에 폭넓게 사용되어온 MPEG-4[10] Simple profile (SP)간의 원활한 통신을 하기위한 화소영역에서의 두 가지 트랜스코딩 방법을 제안 한다. 화소영역에서 트랜스코딩을 하는 이유는 H.264 내에서 비선형적인 (nonlinear) loop filtering을 수행하기 때문에 주파수 영역에서의 트랜스코딩을 이용할 수 없다. 영상의 화질적인 측면에서 가장 좋은 트랜스코딩 방법은 입력된 영상의 비트스트림을 복호화 하고, 복원된 영상을 다시 부호화 하는 직렬 화소영역 트랜스코딩(Cascaded Pixel

* 학생회원, ** 정회원, 세종대학교 인터넷공학과
(Dept. of Internet Engineering, Sejong University, DMS Lab. yllee@sejong.ac.kr)

※ 본 연구는 Tcom & dtvro 위탁연구 지원에 의해 수행되었습니다.
접수일자: 2004년8월10일, 수정완료일: 2004년11월15일

-domain Transcoding)방법이다. 하지만 이 방법은 부호화된 영상을 복호화 하고 복원된 영상을 다시 부호화 하기 때문에 복잡도가 높아지게 된다. 본 논문에서 제안된 두 가지 트랜스코딩 방법은 복잡도(time com-plexity)를 줄이기 위해서 H.264 부호기에서 결정된 각각의 매크로블록(macroblock, MB)의 정보를 재사용하는 트랜스코딩 방법을 제안한다.

본 논문은 다음과 같이 구성된다. Ⅱ장에서는 H.264 BP 와 MPEG-4 SP간의 차이점에 대해 설명하고, Ⅲ장에서는 H.264에서 MPEG-4로 변환 시 화소영역에서 블록타입 간 변환 및 움직임 벡터 조정을 이용한 새로운 두 가지 트랜스코딩방법 대해 소개한다. Ⅳ장에서는 실험 결과와 제안된 알고리듬의 분석이 이루어지며, 마지막으로 5장에서는 결론을 맺는다.

Ⅱ. H.264와 MPEG-4

새로운 동영상 압축 표준인 H.264는 기존의 동영상 압축 표준인 H.261, H.263, MPEG-2 그리고 MPEG-4 들과의 상호 연관성을 고려하지 않고 개발한 표준이기 때문에 영상 압축 방법에 있어 많은 차이점을 보이고 있다. 본 장에서는 표 1에서 나타낸 것과 같이 H.264 BP와 MPEG-4 SP간의 대표적인 차이점에 대해 살펴보면 H.264는 4×4블록단위 정수 DCT(Discrete Cosine Transform)를 수행하는 반면에 MPEG-4는 8×8블록단위 DCT를 한다. 움직임 예측(Motion Estimation, ME) 시 사용하는 H.264의 블록타입과 MPEG-4의 블록타입 또한 상이한 차이를 보이고 있다.

표 1. H.264 Baseline Profile 과 MPEG-4 Simple Profile의 코딩 방법
Table 1. Coding tools of H.264 Baseline Profile and MPEG-4 Simple Profile.

	MPEG-4	H.264
DCT	8×8 DCT	4×4 Integer DCT
MC Unit	16×16, 8×8	16×16, 16×8, 8×16 8×8, 8×4, 4×8, 4×4
MC Accuracy	1/2 pel	1/4 pel
VLC Table	Separable Table	Universal VLC, CAVLC
Intra Prediction	AC/DC Prediction	Spatial Prediction
Inner Loop Filter	None	Deblocking Filter

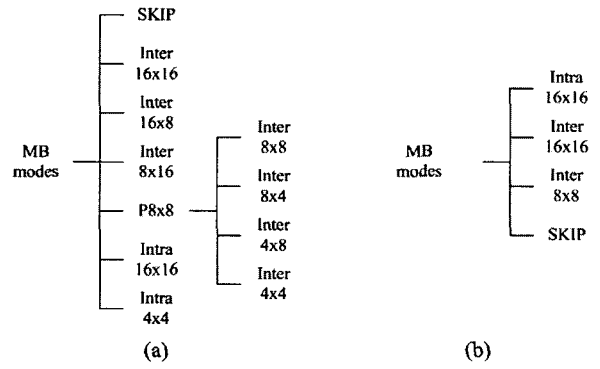

그림 1. (a) H.264의 MB 모드 (b) MPEG-4의 MB 모드
Table 1. (a) MB modes in the H.264 standard, (b) MB modes in the MPEG-4 standard.

그림 1처럼 MPEG-4는 16×16블록 그리고 8×8블록 2가지 블록형태를 가지고 있지만 H.264는 7개의 가변블록에 대한 1/4화소 단위의 움직임 예측 및 움직임 보상(ME/MC)을 한다. 즉, 각각의 16×16 MB을 위한 16×16, 16×8, 8×16 그리고 각각의 8×8 블록을 위한 8×8, 8×4, 4×8, 4×4 단위로 ME/MC를 한다. 각각의 MB내에는 Intra4×4, Intra16×16 그리고 SKIP 모드를 포함한 다양한 Inter 블록 모드를 가지고 있다. 반면에 MPEG-4는 1/2화소 단위의 ME/MC를 한다. 즉, 각각의 16×16 MB를 위한 16×16 그리고 8×8 블록 단위로 ME/MC를 실행한다. 그리고 MPEG-4는 각각의 MB을 위한 Inter16×16, Inter8×8, Intra 그리고 SKIP 모드를 가진다.

Ⅲ. 제안된 두 가지 트랜스코딩 방법

H.264의 비트스트림을 MPEG-4의 비트스트림으로 변환하는 가장 간단하면서도 강력한 방법은 그림 2와 같이 입력으로 들어오는 부호화된 H.264 영상을 모두 복원하여 MPEG-4에서 복원된 영상을 다시 압축하는 직렬 화소영역 트랜스코딩 방식이다. 하지만 이 방법은 모든 프레임에 있는 MB에 대한 움직임 예측 과정이 MPEG-4 부호기에서도 모두 수행하게 된다. 즉 전체

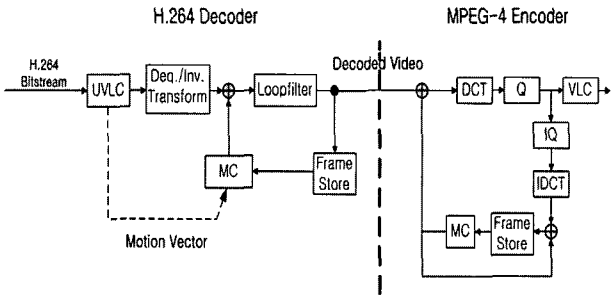

그림 2. 직렬 화소영역 트랜스코딩
Fig. 2. Cascaded pixel-domain transcoding.

변환 시간에 많은 비용을 소비하기 때문에 실시간 전송에도 많은 문제가 뒤따른다.

1. 같은 공간적/시간적 해상도를 유지하는 제안된 트랜스코딩 방법 (방법 A)

위에서 언급한 문제를 해결하기 위해서 움직임 예측 과정을 수행하지 않으면서 실시간 전송에 적합하도록 하기 위하여 H.264 부호기에서 결정된 블록모드 및 움직임 벡터 정보를 MPEG-4 부호기에서 재사용 함 으로써 가능해진다. 그림 3은 H.264 비트스트림을 MPEG-4 비트스트림으로 변환하는 과정을 나타낸 것 이다. 이처럼H.264의 움직임 벡터와 MB 정보를 재사용하기 위해서 H.264가 채택하고 있는 1/4화소 단위의 ME/MC, 7개의 Inter 블록, Intra16×16, Intra4×4, 그리고 SKIP 블록들을 MPEG-4에서 채택하고 있는 1/2 단위의 ME/MC, Inter16×16, Inter8×8, Intra 그리고 SKIP 블록들로 변환해 주어야만 한다.

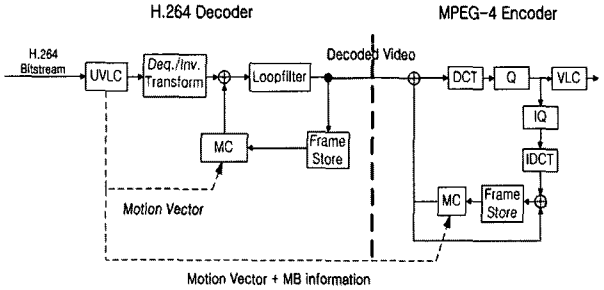

그림 3. 제안된 트랜스코딩 방법
Fig. 3. Proposed transcoding.

그림 4는 H.264의 블록 모드를 MPEG-4의 블록 모드로 변환하는 방법에 대해 설명하고 있다. 그림 4(a)처럼 H.264에서 결정된 Intra4×4와 Intra16×16 모드인 경우에는 MPEG-4에서 사용되는 Intra 모드로 변환한다. H.264에서 결정된 SKIP모드는 그림 4(b)처럼 MPEG-4에서 사용되는 Inter16×16 모드로 변환 한다. 그림 4(c, d)처럼 H.264에서 블록 모드가 Inter16×16 그리고 IntreP8×8 모드로 결정되어진 경우에는 MPEG-4에서

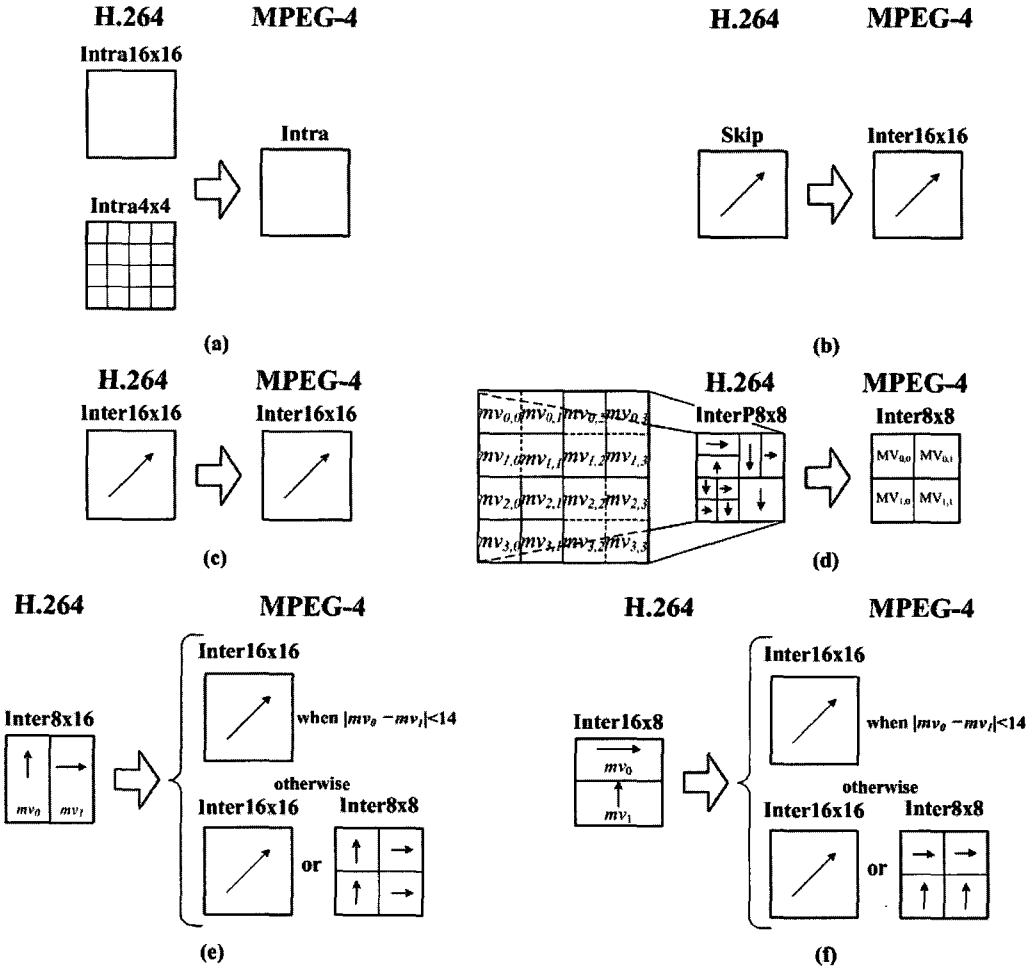

그림 4. 블록 형태 변환 및 움직임 벡터 조정 (방법 A)
Fig. 4. Block type conversion and motion vector mapping (method A).

사용 가능한 Inter16×16 그리고 Inter8×8 모드로 각각 변환 한다. 이때 H.264의 움직임 벡터는 4×4 블록에 대한 1/4화소 단위로 구성되어져 있다. 그림 4(d)에서 설명하는 것과 같이 4×4 블록기반의 1/4화소 움직임 벡터 $mv_{l,k}$는 MPEG-4에서 사용되는 Inter8×8 블록에 대한 정수화소 단위의 움직임 벡터 $MV_{i,j}$로 변환 하여 사용한다.

$$MV_{i,j} = \left(\frac{\sum_{k=2i}^{2i+1} \sum_{l=2j}^{2j+1} mv_{k,l}}{4} + 2 \right) >> 2 \qquad (1)$$

$$i,j = 0,1, \quad k,l = 0,1,2,3$$

위 식의 아래 첨자 i,j는 4개는 8×8 블록들의 세로축, 가로축의 움직임 벡터의 좌표를 가리키고, k,l은 16개의 4×4 블록들의 세로축, 가로축의 움직임 벡터 좌표를 의미한다. H.264에서 결정된 블록 모드가 Inter16×8과 Inter8×16인 경우에는 MPEG-4의 Inter16×16 와 Inter8×8 모드의 중간 모드이기 때문에 양쪽으로의 변환이 가능하다. 이러한 경우에는 그림 4(e, f)에서 보여주는 것과 같이 움직임 벡터 차분 값 비교와 SAD(Sum of the Absolute Differences) 비교를 통한 MPEG-4 부호기에서 사용할 모드를 결정하게 된다. 즉, H.264의 Inter16×8 또는 Inter8×16 모드로 구성된 MB의 1/4 화소단위의 움직임 벡터의 차분 값$|mv_0 - mv_1|$이 실험에 의해서 결정된 문턱값 (threshold) 14 보다 작으면 Inter16×16 모드로 변환하고, 그렇지 않은 경우에는 Inter16×16 블록의 SAD값(SAD-128)과 4개의 Inter8×8 블록 각각의 SAD값들의 합을 비교해서 최소 SAD값을 가지는 모드를 선택한다. 이 과정을 통해서 부호화 되는 비트의 양을 감소시킬 수 있다.

2. 영상 축소를 이용한 제안된 트랜스코딩 방법(방법 B)

영상 축소를 이용한 제안된 트랜스코딩은 그림 3에서 보여주는 H.264 복호기와 MPEG-4 부호기사이에 down-sampling filter를 첨가함으로써 실행되어 진다. 그림 5는 H.264 복호기 에서 복원된 CIF(Common Intermediate Format) 영상의 각 2×2 MB(4개의 MB)을 MPEG-4 부호기에서 부호화할 한 개의 MB으로 줄여 QCIF(Quarter Common Intermediate Format) 영상으로 만드는 과정을 보여준다. CIF 영상을 QCIF 영상으로 축소시킬 때 4개의 화소들을 average filter를 사용하여 1개의 화소로 sampling하는 방법 그리고 4개의 화소들 중 1개의 화소만을 취하는 sub-sampling 기법이

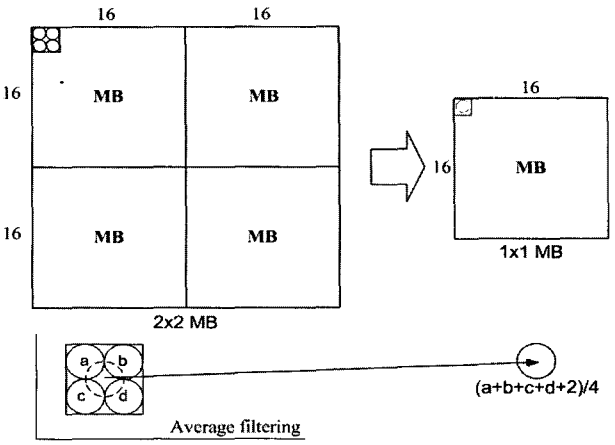

그림 5. Average filter를 이용한 영상 축소
Fig. 5. Down-sampling by using average filter.

있다. 본 논문에서는 average filter를 이용하여 영상을 축소시키는 방법을 제안한다. 그 이유는 sub-sampling 기법을 이용하여 영상을 축소시키면 시간 복잡도적인면에서 약간의 이득은 볼 수 있지만 영상의 화질저하가 현저히 나타나기 때문에 average filter를 이용하였다.

그림 6에서는 복원된 H.264 비트스트림의 2×2 MB들의 형태(MB mode)를 MPEG-4에서 재사용할 수 있는 하나의 MB 형태로 변환해 주는 방법을 제안한다. H.264내의 모든 Skip 모드는 H.264내의 2×2 MB들을 MPEG-4내의 한 개의 MB으로 변환하기 위한 전처리 단계로써 그림 6(a)와 같이 H.264내의 Inter16×16 모드로 변환 시킨다. 그림 6(b)처럼 H.264의 2×2 MB들이 모두 Intra 모드(Intra16×16, Intra4×4)인 MB들로만 구성되어 있으면 한 개의 Intra16×16 MB으로 변환하고 average filter를 이용하여 영상 축소를 한다. 그림 6(c)처럼 2×2 MB들이 2개 또는 3개의 Intra 모드 MB들로 구성되어 있는 경우, MPEG-4 부호기에서 Inter16×16, Inter8×8 그리고 Intra 16×16 모드 중 어떤 모드로 변환해주는 기준이 불명확한 경우이다. 이런 경우 MPEG-4 내에 있는 특정한 하나의 모드(Inter16×16, Inter8×8 또는 Intra16×16 모드)로 변환 하였을 경우 화질이 떨어지게 되고 부호화된 비트양이 증가하는 현상이 나타난다. 그래서 시간적인 손해를 보더라도 MPEG-4 부호기내에서 Inter16×16 블록의 SAD값(SAD-128)과 4개의 Inter8×8 블록 각각의 SAD값들의 합을 비교해서 최소 SAD값을 가지는 모드를 선택하고, 선택되어진 Inter 모드 SAD값과 다음과 같이 구해진 Intra 모드 SAD값을 비교해서 최소 SAD값을 가지는 모드를 최종적으로 선택한다.

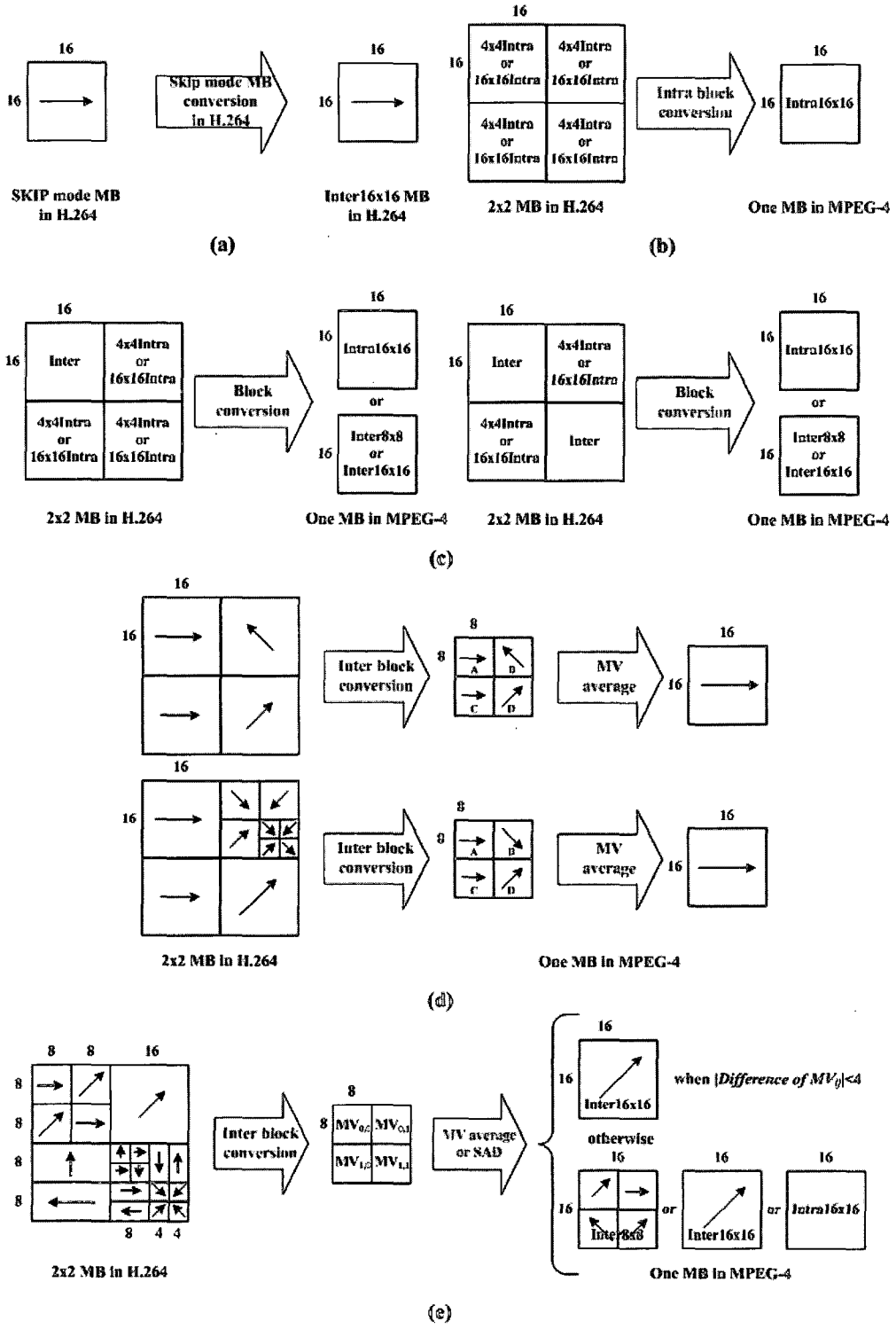

그림 6. 블록 형태 변환 및 움직임 벡터 조정 (방법 B)

Fig. 6. Block type conversion and motion vector mapping (method B).

$$MB_{mean} = \sum_{i=0}^{15}\sum_{j=0}^{15} org_{i,j}$$

$$SAD_{Intra} = \sum_{i=0}^{15}\sum_{j=0}^{15} | org_{i,j} - MB_{mean} | \qquad (2)$$

위 식에서 $org_{i,j}$는 MB내에 있는 각각의 16×16 화소들의 값이고 MB_{mean} 은 MB내에 있는 화소 값들의 평균이다. 이와 같이 과정을 거쳐 선택된 최적의 모드로 변환시킴으로써 화질적인 측면이나 부호화되는 비트의 량을 줄이는 이득을 볼 수 있다. 그림 6(d)의 경우와 같이 H.264의 2×2 MB들내에 Inter16×16 모드가 4개 또는

3개가 있는 경우 MPEG-4 부호기내의 Inter16×16 또는 Inter8×8 모드로 변환이 가능하다. 하지만 본 논문에서는 Inter8×8 모드가 아닌 Inter16×16 모드로 변환한다. 즉, 그림 6(d)와 식(3)에서 설명하는 것과 같이 우선 2×2 MB들내의 각각의 MB의 1/4단위 움직임 벡터의 평균값을 MPEG-4 MB내의 4개의 8×8 블록들의 정수단위 움직임 벡터로 취한 후 이렇게 얻은 4개의 8×8 블록 움직임 벡터의 평균값으로 하나의 움직임 벡터를 Inter16×16 MB에 취함으로써 모드 변환이 이루어진다. 이와 같이 Inter8×8 모드가 아닌 Inter16×16 모드로 변환하는 이유는 H.264에서 MPEG-4로 트랜스코딩 시 foreman 영상에 대해 QP(quantization parameter)를 10으로 고정한 후 모드 변환에 대한 모든 조건을 동일하게 하고 위의 경우에 대해서만 Inter8×8 모드로 변환하면 Inter16×16 모드로 변환 하였을 때 보다 PSNR이 31.61dB에서 31.69dB로0.08dB 향상을 보였지만 부호화된 비트양은 143.2kbps에서 156.8kbps로 증가하는 현상을 보이기 때문에 전체적이 부호화 효율이 떨어지기 때문이다. 그림 (e)와 같이 H.264의 2×2 MB들이 복잡한 블록모드 즉, 한 개 이상의 InterP8×8 모드인 MB들, 한 개 이하의 Intra 모드 MB 그리고 이들 외의 Inter 모드 MB들로 구성되어 있는 경우, 2×2 MB들내의 각각의 MB을 MPEG-4내의 8×8 블록으로 변환 시킨다. MPEG-4내의 8×8 블록이 취하는 정수단위 움직임 벡터 $MV_{i,j}$ 는 16개의 4×4 블록이 가지고 있는 1/4단위 움직임 벡터 $mv_{m,n,k,l}$의 평균값을 구하고 이 값을 다시 4로 나눔으로써 정수단위의 움직임 벡터를 구한다. 계산식은 다음과 같다.

$$MV_{i,j} = \left(\frac{\sum_{k=0}^{3}\sum_{l=0}^{3} mv_{m,n,k,l}}{\sum_{k=0}^{3}\sum_{l=0}^{3} 1} + 2 \right) >> 2 \qquad (3)$$

$$i,j = 0,1, \qquad k,l = 0,1,2,3$$

위 식의 i,j는 중간 단계인 MB이 취하는 8×8 블록들의 가로축, 세로축 좌표를 가리키고 k,l 은 2×2 MB들 안의 (m, n)번째 MB이 취하는 4×4 블록들의 가로축, 세로축 좌표를 가리킨다. MPEG-4 블록 형태를 위한 모드결정은 다음 단계를 수행하면서 이루어진다. 만약 모든 정수단위 움직임 벡터 $MV_{i,j}$ 간의 차분 값이 4보다 작으면 Inter16×16 MB 모드로 변환하고 그렇지 않으면 그림 6(c)에서 언급한 것과 같이 SAD 과정을 통한 MB 모드를 결정하게 된다. H.264 부호기에서 Inter8×8 , Inter8×4, Inter4×8 그리고 inter4×4 모드처럼 edge로 인하여 결정된 모드를 MPEG-4 부호기에서 사용할 모드를 Inter8×8 MB모드가 아닌 Inter16×16 MB 모드로 변환 하는 이유는 H.264로 부호화 및 영상 축소 과정이 일종의 LPF(Low Pass Filtering)이기 때문이다.

3. 움직임 벡터 재조정(motion vector refinement)

트랜스코딩의 효율을 향상시키기 위해서 식 (1)과 식 (3)에 의해서 생성된 완전하지 않은 정수단위 움직임 벡터를 중심으로 움직임 벡터 재조정을 한다. 같은 공간적, 시간적 해상도를 유지하는 제안된 트랜스코딩(방법 A)에서는 새로 찾은 움직임 벡터 중심으로 ±1개의 이웃화소(neighbor pixel)에 대하여 정수단위 움직임 벡터를 찾고, 찾은 움직임 벡터 주변으로 1/2화소 단위의 이웃화소에 대해 ±1개 움직임 벡터를 찾음으로써 움직임 벡터 재조정을 한다. 또한 제안된 영상축소를 이용한 트랜스코딩(방법 B)에서는 새로 찾은 움직임 벡터 중심으로 방법 A에서 보다 많은 ±3개의 이웃화소(neighbor pixel)에 대하여 정수단위 움직임 벡터를 찾고, 찾은 움직임 벡터 주변으로 1/2화소 단위의 이웃화소에 대해 ±1개 움직임 벡터를 찾음으로써 움직임 벡터 재조정을 한다. 이와 같은 움직임 벡터 재조정을 통하여 트랜스코딩의 효율을 높일 수 있다.

Ⅳ. 실 험

본 실험은 H.264 BP JM(Joint Model) 7.3 Decoder[11]와 MPEG-4의 SP MoMuSys-FDIS-V1.0 Encoder[12]를 기반으로 만든 Transcoder를 이용하여 실험하였다. 실험은 펜티엄 IV 2.66GHz PC에서 실험 하였다. 제안된 트랜스코딩 알고리듬의 성능 평가를 위해서 8개의 비디오 영상 즉, 트랜스코딩 방법 A를 위한 QCIF 크기의 Foreman, Coast, News, Paris 영상들과 트랜스코딩 방법 B를 위한 CIF 크기의 Foreman, Coast, News, Paris 영상들에 대한 PSNR, Bitrates 그리고 트랜스코딩 수행시간에 대해 분석 하였다. 제안된 트랜스코딩 방법 A에서는H.264 부호기에서 300장의 영상을 10Hz의 프레임 율로 압축한 비트스트림을 실험 A에서 사용 하였으며, 제안된 트랜스코딩 방법 B에서는 H.264 부호기에서 300장의 영상을 30Hz의 프레임 율로 압축한 비트스트림을 사용하였다. 그리고 두 개의 실험에서 각각의 비디오 영상들은 첫 번째만 INTRA 프레임이고 나머지는

표 2. 각 영상에 대한 시간 복잡도 비교 (Proposed transcoding algorithm A)
Table 2. Comparisons of the computation time for each sequence (Proposed transcoding algorithm A).

	Cascade transcoding algorithm			Proposed transcoding algorithm A		
QCIF Sequences	H.264 Decoder time	MEPG-4 Encoder time	Total time	H.264 Decoder time	MEPG-4 Encoder time	Total time
Foreman	5.0	14.17	19.17	5.0	2.96	7.96
Coast	5.25	16.23	21.48	5.25	3.09	8.34
News	4.23	14.78	19.01	4.23	2.78	7.01
Paris	4.59	15.79	20.38	4.59	2.82	7.41

표 3. 각 영상에 대한 시간 복잡도 비교 (Proposed transcoding algorithm B)
Table 3. Comparisons of the computation time for each sequence (Proposed transcoding algorithm B).

	Cascade transcoding algorithm			Proposed transcoding algorithm A		
QCIF Sequences	H.264 Decoder time	MEPG-4 Encoder time	Total time	H.264 Decoder time	MEPG-4 Encoder time	Total time
Foreman	49.54	53.49	103.03	49.54	13.32	62.86
Coast	51.48	53.54	105.02	51.48	13.29	64.77
News	34.14	52.23	86.37	34.14	12.1	46.24
Paris	37.2	52.18	89.38	37.2	12.3	49.5

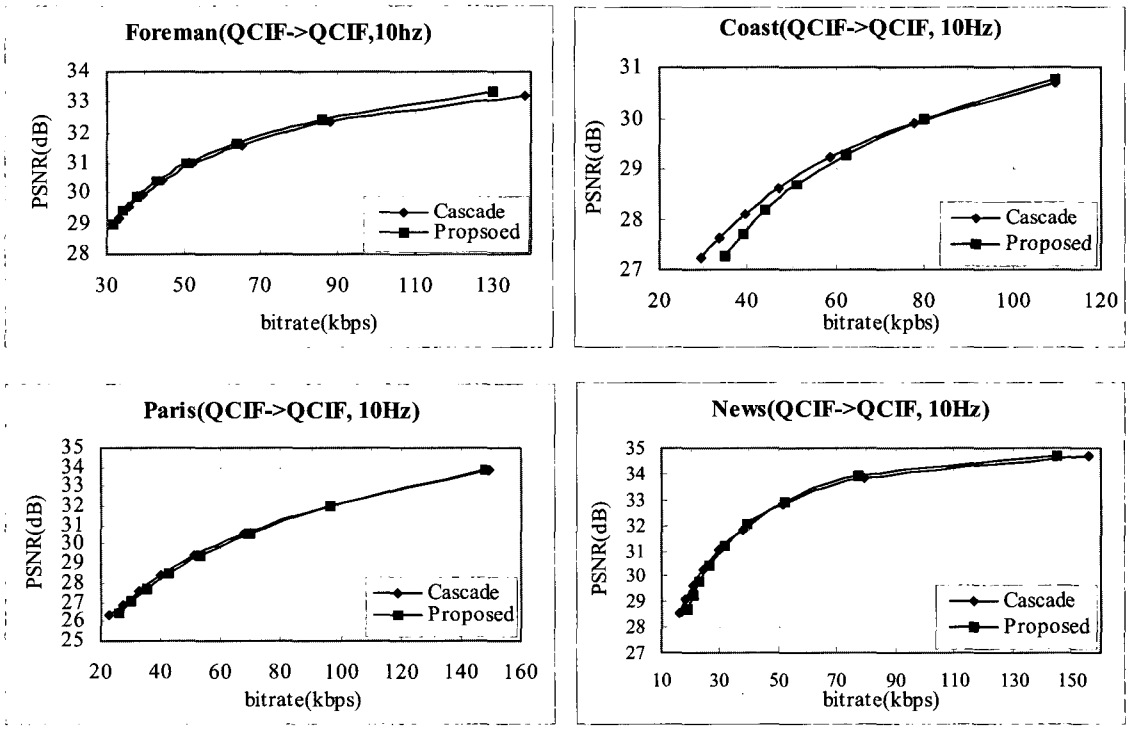

그림 7. 직렬 화소영역 트랜스코딩 방법과 제안된 트랜스코딩 방법 A 간의 PSNR 비교
Fig. 7. Comparison of the PSNR between the cascade transcoding method and the proposed transcoding method A.

모두 B 프레임을 제외한 INTER 프레임 즉, "I, P, P, P.."의 구조를 가진다. 표 2, 3에서는 제안된 두 가지 트랜스코딩 방법으로 트랜스코딩 하였을 경우 직렬 화소영역 트랜스코딩 방법 보다 얼마만 큼의 시간적 이득을 보았는지를 보여주고 있다. 각각의 영상들 마다 약간의 차이는 보이지만 직렬 화소영역 트랜스코딩 방법보다 두 개의 제안된 트랜스코딩 방법들이 전체 트랜스코딩 시간을 고려했을 때 약 1.7~2.6배 정도의 속도향상을 보였고, MPEG-4 부호기에서의 수행 시간만을 고려했을 시 에는 4.1~5.1배 정도의 속도 향상을 보였다. 그림 7,

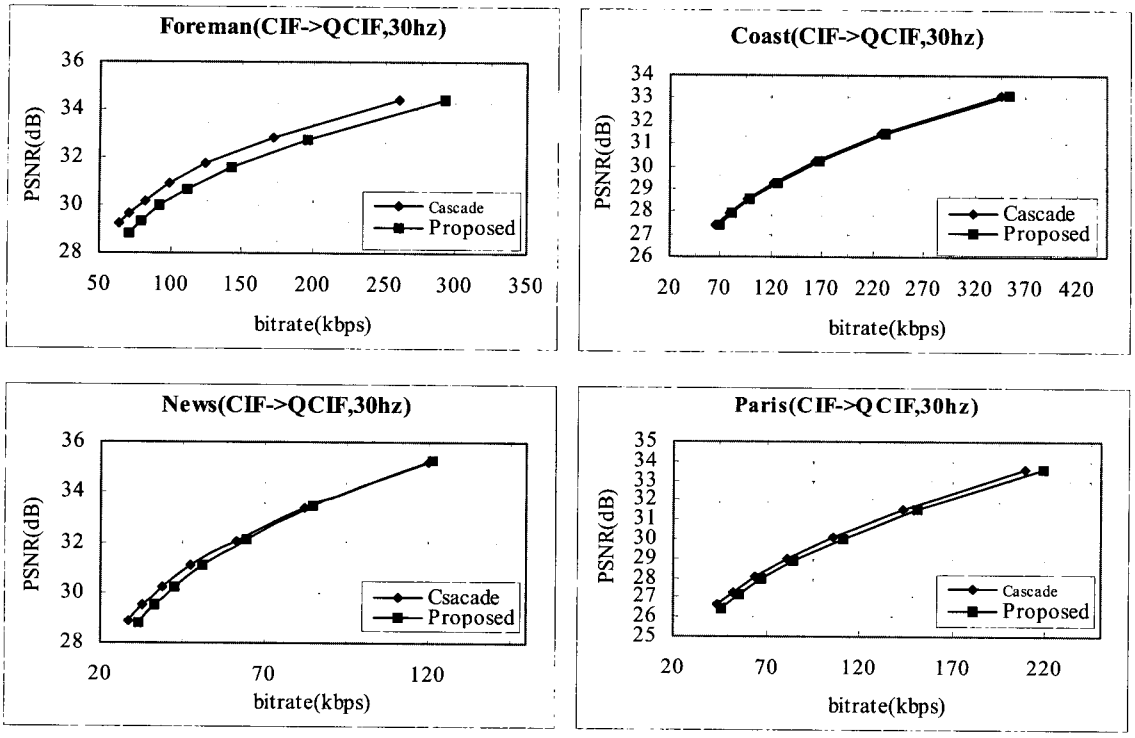

그림 8. 직렬 화소영역 트랜스코딩 방법과 제안된 트랜스코딩 방법 B 간의 PSNR 비교
Fig. 8. Comparison of the PSNR between the cascade transcoding method and the proposed transcoding method B.

8은 Foreman, Coast, News 그리고 Paris 영상들을 각각 H.264에서 MPEG-4로 변환 시 직렬 화소영역 트랜스코딩 방식, H.264에서 MPEG-4로 변환 시 본 논문에서 제안한 두 가지 트랜스코딩 방식으로 영상을 압축했을 때 밝기(Y)신호의 PSNR의 차이를 보여주고 있다. 그림 7에서 보여주는 것과 같이 제안된 트랜스코딩 방법 A의 PSNR은 낮은 bitrates 영역에서는 직렬 화소영역 트랜스코딩의 PSNR보다 0.1~0.3dB 정도의 화질이 저하 했지만, 높은 bitrates 영역에서는 모든 영상에 대해서 0.1~0.15dB 정도의 화질이 개선되었다. 그리고 특별하게 Foreman 영상에서는 모든 bitrates 영역에서 직렬 화소영역 트랜스코딩보다 0.1~0.2dB 정도의 화질이 개선되었다. 제안된 트랜스코딩 방법B의 PSNR은 그림 8에서 보여주는 것과 같이 직렬 화소영역 트랜스코딩의 PSNR보다 0.1~0.5 정도의 화질 저하를 관찰할 수 있었다.

V. 결 론

본 논문에서는 4×4 블록 기반의 H.264의 움직임 벡터와 H.264의 MB정보를 재사용함으로써 낮은 복잡도를 가지는 H.264와 MPEG-4 이기종간의 트랜스코더를 제안하였다. 시간적으로 많은 비용 갖는 직렬 화소영역 트랜스코딩 방식에 비해 H.264 복호기 에서 복원된 움직임 벡터와 블록정보를 재사용함으로써 시간적으로 1.7~2.6배 정도의 속도향상을 보이면서 화질열화가 거의 일어나지 않는 트랜스코딩 방법이 제안되었다. 이와 같이 H.264에서 MPEG-4로 트랜스코더를 개발함으로써 기존의 MPEG-4코덱을 사용했던 장비들을 교체하지 않고도 새로운 동영상 압축 표준인 H.264로 부호화된 영상을 이용가능하게 할 것이다.

참 고 문 헌

[1] J. Jeongnam Youn, M.T. Sun, and C.W. Lin, "Motion Vector Refinement for High Perfor-mance Transcoding," IEEE Trans. Multimedia, vol.1, no.1, pp.30~40, March 1999.

[2] A. Vetro, C. Christopoulos, and H. Sun,"Video Transcoding Architectures and Techniques: An Overview," IEEE Signal Processing Magazine, March 2003. pp18-29.

[3] P. Yin, M. Wu, and B. Lui and H. Sun, "Drift compensation for reduced spatial resolution transcoding," IEEE Trans. Circuits Syst. Video Technol., vol. 12, pp.1009-1020, Nov. 2002.

[4] P. Yin, M. Wu, and B. Lui, "Video transcoding by reducing spatial resolution," in Proc. IEEE Int. Conf. Image Processing, Vancouver, BC, Canada, vol. 1, Oct. 2000, 972-975.
[5] P. Assuncao and M. Ghanbari, "A frequency Domain video transcoder for dynamic bit-rate reduction of MPEG-2 bitstreams," IEEE Trans. Circuits Syst. Video Technol. vol. 8, pp.953 ~ 967,Dec. 1998.
[6] C. W. Lin and Y. R. Lee, "Fast algorithms for DCT-domain video transcoding," in Proc. IEEE Int. Conf. Image Processing, Thessaloniki, Greece, vol. 1, Sept. 2001, pp.421-424.
[7] D. G. Morrison, M.E. Nilsson, and M. Ghanbari, "Reduction of bit-rate of compressed video while in its coded form," in Proc. 6th Int Workshop on Packet Video, Portland, OR, Sept. 1994, pp.392 -406.
[8] H. Sun, W. Kwok, and J. W. Zdepski, "Architec -tures for MPEG compressed bitstream scaling," IEEE Trans. Ciruits Syst. Video Technol., vol.6, pp. 191-199, Apr. 1996.
[9] Tomas Wiegand, Joint Final Committee Draft (JFCD) of joint Vide specification(ITU-T Rec. H.264 |ISO/ICE 1449-10 AVC), JVT-G050, March 2003.
[10] Weiping Li, Jens-Rainer Ohm, Mihaela van der Schaar, Hong Jiang, Shipeng Li, "MPEG-4 Video Verification version 17.0", ISO|IEC JTC1|SC29|WG11 N3515, July 2000.
[11] http://bs.hhi.de/~suehring/tml/download/old_jm /jm73.zip
[12] MPEG ftp site.

저 자 소 개

권 혁 균(학생회원)
2003년 세종대학교 응용화학과 학사
2003년~현재 세종대학교 대학원 인터넷공학과 석사 과정
<주관심분야: 영상처리, Video Transcoding, Watermaking>

이 영 렬(정회원)
1985년 서강대학교 전자공학과 학사
1987년 서강대학교 전자공학과 석사
2000년 한국과학기술원 전기 및 전자공학과 박사
2001년 8월 삼성전자 중앙연구소 DMS Lab. 수석 연구원
2001년 9월~현재 세종대학교 인터넷공학과 부교수
<주관심분야: 영상처리, 멀티미디어 시스템, 비디오트랜스 코딩, 3차원, 스케일러블 비디오코딩>