논문 2004-41SP-3-4

DCT 코덱에 기반한 변환 영역에서의 리사이징 알고리즘

(Transform Domain Resizing for DCT-Based Codec)

신 건 식*, 장 준 영*, 강 문 기**

(GunShik Shin, JoonYoung Jang, and Moon Gi Kang)

요 약

공간 영역에서의 처리되는 것과 똑같은 효과를 주는 변환 영역에서의 영상 처리 방법은 영상의 효율적인 전송에 중요하다. 이 논문에서는 DCT(discrete cosine transform) 영역에서 영상의 크기를 늘이고 줄이는 리사이징 방법을 다루고, 리사이징을 변환 영역에서 처리했을 경우의 효과를 공간영역에서의 리사이징과 비교하여 분석한다. 그리고 이 분석에 기초하여 두 가지 리사이징 알고리즘을 제안한다. 첫 번째 알고리즘은 영상을 압축하기 전에 영상의 크기를 줄여서 영상의 압축률을 높이는 방법이고, 두 번째 알고리즘은 기존의 압축율을 유지하면서 영상 정보의 손실을 줄이는 방법이다. 이 알고리즘은 표준 코덱들과의 호환성을 가지고 있기 때문에, 널리 사용되고 있는 코덱과 쉽게 결합될 수 있다. 뒤에 제시되는 실험 결과에서는 동등한 화질을 가지면서도 영상의 비트 수는 반으로 줄어드는 것과, 똑같은 압축 비율에서 영상의 화질이 2~3dB 정도 개선되는 것을 확인할 수 있다.

Abstract

The ability to perform the same operations in the transform domain as in the spatial domain is important for efficient image transmission through a channel. We perform image resizing, which includes magnifying and reducing the size, in the discrete cosine transform(DCT) domain and the effects of the transform domain approach are analyzed in the corresponding spatial domain. Based on this analysis, the two resizing algorithms are proposed. The first one further compresses the images encoded by the compression standard by reducing the size before compression, and the other reduces the loss of information while maintaining the conventional compression rate. Because of its compatibility with standard codec, these algorithms can be easily embedded in JPEG and MPEG codecs, which are widely used for the purpose of image storage and transmission. Experimental results show a reduction of about half the bit size with similar image quality and about a 2- or 3-dB quality improvements in the similar compression rate.

Keywords: resizing; image compression; discrete cosine transform; transform domain processing; restoration.

I. 서 론

최근의 영상 압축 코덱인 JPEG이나 MPEG 1,2,4 /H.261, 263, 264와 같은 정지영상 및 동영상 압축에 쓰이는 많은 방법들은 DCT(discrete cosine trans -form)에 기반한 압축 알고리즘을 사용한다. 영상을 다루는 다양한 분야에서는 앞에서 언급한 압축 규격에 따라 압축된 영상의 상태로, 리사이징(영상 크기의 확대와 축소), 필터링, 움직임 보상 등의 처리를 해야 하는 경우가 생긴다. 이중에서 영상 해상도의 변화는 다양한 네트워크 환경을 가지는 사용자의 요구를 만족하기 위한 비디오 트랜스코딩(transcoding)에서 자주 사용되어지는 방법이다[1][2].

그러나 공간영역에 기반한 리사이징 방식을 사용하는 경우, 리사이징을 하는 데 여러 가지 문제점들이 존재한다. 우선 첫 번째로 영상의 리사이징을 위해서는 압축(compression)과 복원(decompression)을 해야 하기 때문에 불필요한 계산량이 증가한다. 일반적인 리사이징 기법은 압축된 영상을 역변환을 통해 복원된 후, 리사이징을 하고 필요에 따라서 다시 압축을 해야 하는데[3], 이것은 시스템의 중복으로 계산량 증가를 초래함으

학생회원, **정회원, 연세대학교 전기전자공학과
(Dept. of Eletrical and Electronic Engineering, Yonsei Univ.)
접수일자: 2004년2월21일, 수정완료일: 2004년4월14일

로써 실시간 처리가 힘들어진다. 두 번째로 일반적으로 이러한 변환 역변환 과정은 손실 시스템이기 때문에 불필요한 변환과 역변환을 계속 반복하면 할수록 원래의 영상 정보가 손실되는 문제점도 가지고 있다. 세 번째로, 기본적인 리사이징 알고리즘은 공간 영역에 기반하므로 변환 영역에서만 처리하는 것이 아니라, 공간 영역에서도 영상을 담을 메모리가 필요하게 된다. 이것은 추가 메모리를 필요로 하게 되어 시스템의 크기를 크게 한다.

이러한 여러 문제점들을 극복하기 위해 불필요한 압축과 복원 과정을 피하고 공간 영역의 리사이징 방법에 의존하지 않고, 변환 영역에서 바로 리사이징을 할 수 있는 알고리즘을 개발하게 되었다. 이 논문에서는 변환 영역에서의 리사이징 방법이 제안되어 있다. 이 방법을 사용함으로써 시스템의 복잡성을 해결할 수 있을 뿐만 아니라 데이터 전송에서의 더 나은 압축 효율을 얻음과 동시에 열화 현상도 획기적으로 줄일 수 있다.

Ⅱ장 1절에서는 변환영역에 기반한 방법 중 공간 리사이징 필터를 사용했을 때의 문제점을 보였고, Ⅱ장 2절에서는 변환영역에 기반한 방법의 이론적인 분석을 보였다. Ⅱ장 3절에서는 이에 대한 구현을 다뤘으며, Ⅲ장에서는 실험 결과를 보였다. 마지막으로 Ⅳ장에서는 결말으로 끝을 맺었다.

Ⅱ. 본 론

1. 변환 영역에서의 리사이징

변환 영역에서의 필터링 개념은 제일 처음 [4]에서 제안되었다. 이 논문에서, 저자는 변환 행렬과 공간 필터링 행렬을 하나의 행렬로 통합하였다. 이 밖에도 [3], [5]에서는 DCT 영역에서의 리사이징 방법이 제안되어 있다. 하지만 이러한 방법은 공간 영역에서의 필터링 연산을 바탕으로 이루어졌다.

지금부터는 DCT 영역에서 바로 리사이징을 수행하는 방법을 알아보겠다. 행렬 X와 Y는 행이나 열을 쌓아 만든 벡터가 아니라 영상 자체라고 하자. 이 때 $N\times N$크기의 블록 행렬 Y는 $2N\times 2N$ 블록 행렬 X로부터 생성된 것이라 하면, 영상의 크기를 줄이는 리사이징을 다음과 같이 표현할 수 있다.

$$Y = HXH^T \tag{1}$$

H는 임의의 리사이징 커널을 의미한다. (1) 식에 DCT의 배분 법칙을 적용시키면 다음과 같이 된다.

$$DCT(Y) = DCT(H)DCT(X)DCT(H^T) \tag{2}$$

[1]과 [3]에서는 다음과 같은 $N\times 2N$ 크기의 H 행렬이 사용되어졌다.

$$H = \begin{matrix} H_1 & \Phi \\ \Phi & H_1 \end{matrix} \tag{3}$$

여기서 Φ는 $N/2\times N$ 크기의 영행렬이고, 만약 H_1 행렬이 $N=8$일 때 다음과 같은 모양을 가진다.

$$H_1 = \begin{matrix} 0.5 & 0.5 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0.5 & 0.5 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0.5 & 0.5 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0.5 & 0.5 \end{matrix} \tag{4}$$

이 행렬은 이단(two-tab) 필터를 나타내는 행렬로서 주파수 특성은 좋지 않지만 경계효과를 발생시키지 않는다. 이에 반해 선형(bilinear) 필터와, 고차 라그랑즈(Lagrange) 필터의 경우 경계효과를 발생시키지만 좋은 주파수 특성을 가진다. 다른 필터 H_b, H_c는 다음과 같은 모양을 가진다.

$$H_b = \begin{matrix} 0.5 & 0.25 & 0 & 0 & 0 & 0 & 0 & 0.25 \\ 0 & 0.25 & 0.5 & 0.25 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0.25 & 0.5 & 0.25 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0.25 & 0.5 & 0.25 \end{matrix} \tag{5}$$

$$H_c = \begin{matrix} 0.5 & 0.281 & 0 & -0.031 & 0 & -0.031 & 0 & 0.281 \\ 0 & 0.281 & 0.5 & 0.281 & 0 & -0.031 & 0 & -0.031 \\ 0 & -0.031 & 0 & 0.281 & 0.5 & 0.281 & 0 & -0.031 \\ 0 & -0.031 & 0 & -0.031 & 0 & 0.281 & 0.5 & 0.281 \end{matrix} \tag{6}$$

H_b, H_c는 각각 선형(bilinear), 3차 스플라인(cubic cardinal spline) 리사이징 필터를 나타낸다[6]. 행렬 (5), (6)에서 볼 수 있듯이 더 나은 주파수 특성을 가질수록, 경계효과도 더 많이 일어나는 것을 알 수 있다. 그러나 이러한 방법들은 이미 앞에서 언급했듯이 경계효과와 나쁜 주파수 특성을 가지는 공간영역의 리사이징 방법에 그 기반을 두고 있다. 이러한 어려움을 피하기 위해서 변환 영역과 공간 영역을 거치지 않고 직접적으로 리사이징하는 방법이 필요하다. DCT 영역에서의 처리는 공간 영역에서의 필터링보다 적은 경계 효과를 가진다. 왜냐하면 공간영역에서의 필터링이 순환 컨볼루션(circular convolution)으로 이루어지는 반면에 DCT는 대칭 컨볼루션(symmetric convolution)으로 이루어지기 때문이다[7]. 대칭의 특성은 순환 컨볼루션과는 다르게 의도하지 않은 고주파 성분이 생기는 것을 막아준다.

지금부터는 새로운 명명법을 도입한다. 그림 1에는 앞으로의 내용 전개에 필요한 새로운 용어를 정의해 놓

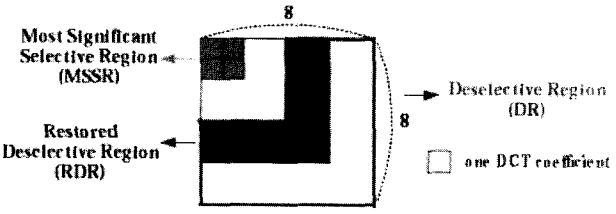


그림 1. 명명법
Fig. 1. Nomenclature

았다. 여기부터 우리는 가장 중요한 선택 영역(most significant selective region)을 MSSR이라고 부를 것이다. 그리고 3. 나. 장에서 설명할 '확장 알고리즘'에서 고주파 성분을 복원하기 위해 선택되어지는 영역(restored deselective region)을 RDR이라고 하고, 압축 시 버려지는 영역(deselective region)을 DR이라고 부를 것이다.

2. 이론적인 분석

DCT 영역에서 영상의 크기를 줄이기 위해서는 다음의 과정이 필요하다. 예를 들어 한 개의 8×8 블록을 4×4크기의 블록으로 줄이기 위해서는 8×8 블록의 고주파 부분인 DR 영역의 48개의 계수들을 버리고 16개의 저주파 부분만을 선택해야 한다. 마찬가지로 영상의 크기를 늘리기 위해서는 4×4 크기의 블록을 8×8블록의 저주파 영역에 위치한 후 나머지 고주파 부분(DR 영역)에 '0'을 넣어주어야 한다(zero-padding). 지금부터는 이러한 리사이징 과정을 DCT 영역에서 했을 때, 이와같은 처리가 공간 영역에서는 어떤 효과를 주는지 설명하도록 하겠다.

가. 영상의 크기를 줄일 경우

이 부분에서 우리는 여러 가지 DCT 타입이 정의되어 있는 [7]에 나와 있는 표기법을 따르도록 하겠다. 첫째로 입력 값을 변환시킨 DCT 계수 $X_{II}(k)$가 주어진다고 하자.

$$X_{II}(k) = C_{IIE}[x(n)] = \sqrt{\frac{2}{N}}\gamma_k \sum_{n=0}^{N-1} x(n)\cos\left[\frac{\pi k(n+1/2)}{N}\right] \quad (7)$$

$k=0$일 때, $\gamma_k = 1/\sqrt{2}$ 이고 그 밖의 경우에는 $\gamma_k = 1$이다. C_{IIE}는 영상 압축 표준에서 가장 많이 사용되어지는 타입-2 DCT이다. 예를 들어 L단(tab) 필터 $h(n)$, $n = -L/2, \cdots, L/2-1$을 생각해보자. 대칭 컨볼루션의 경우, 우리는 다음과 같이 정의된 오른편 반쪽 필터(filter-right-half)가 필요하다.

$$h^r(n) = \begin{cases} h(n) & n = 0,1,...,L/2-1 \\ 0 & n = L/2,...,N-1 \end{cases} \quad (8)$$

공간영역에서의 대칭 컨볼루션은 DCT 영역에서의 컨볼루션-곱셈 특성으로 정의된다. 신호를 어떻게 대칭 확장시키느냐에 따라서 40가지의 대칭 컨볼루션이 가능하다. 모든 신호들은 다음과 같은 4가지 방법에 의해서 대칭성을 가진 신호로 확장된다. 온샘플 대칭(whole-sample symmetry, WS), 온샘플 역대칭(whole-sample antisymmetry, WA), 반샘플 대칭(half-sample symmetry, HS), 반샘플 역대칭(half-sample antisymmetry, HA). 대칭 컨볼루션의 자세한 내용은 [8]을 참고하기 바란다. $h^r(n)$의 타입-2 DCT의 컨볼루션 형태는 다음과 같다.

$$H^r(k) = C_{2e}[h^r(n)] = 2\sum_{n=0}^{N-1} h^r(n)\cos\frac{\pi k(n+1/2)}{N} \quad (9)$$

DCT 영역에서의 곱셈은 공간 영역에서의 필터링과 같다. 입력 신호의 DCT 계수 $X_{II}(k)$와 L단 필터의 DCT 계수 $H^r(k)$로부터 $Y_I(k)$를 다음과 같이 표현할 수 있다.

$$Y_I(k) = \begin{cases} H^r(k)X_{II}(k) & k = 0,1,...,N-1 \\ 0 & k = N \end{cases} \quad (10)$$

$Y_I(k)$에 대응되는 공간영역에서의 신호 $w(n)$은 다음과 같다[2].

$$w(n) = \frac{1}{k(n)} C_{IE}^{-1}[Y_I(k)] \quad (n,k = 0,1,...,N) \quad (11)$$

$$= x(n)\langle_{HSHS}SC_{HSHS}\rangle h^r(n) \quad (12)$$

여기서 C_{IE}^{-1} 은 타입-1의 IDCT를 의미한다. (11)식의 가중치 행렬 $k(n)$은 $n=0$ 일 때와, $n=N$일 때만 $w(n)$에 영향을 미친다. 따라서 다음 단계가 홀수 번째 샘플만을 취하는 다운샘플링이라면 $k(n)$은 아무 작용을 하지 않는다. 여기서 역변환이 타입-2

IDCT인 C_{IIE}^{-1}이 아니고 C_{IE}^{-1}인 것에도 주목해야 한다. 만일 C_{IIE}^{-1}가 사용되어진다면 대칭 컨볼루션에 의한 컨볼루션-곱셈의 특성은 더 이상 맞지 않게 되고, 복잡한 세 번째 항이 필요하게 된다[9]. 다음의 DCT 영역에서의 계산은 공간 영역에서의 다운샘플링의 효과를 얻게 한다[2].

$$Y_d(k) = \frac{Y_I(k) - Y_I(N-k)}{\sqrt{2}} \quad (13)$$

$(k = 0, 1, \cdots, N/2-1)$

$Y_d(k)$를 타입-2 IDCT를 통해, 역변환 하면 다음과 같이 된다.

$$y_d(n) = y(2n+1) = C_{IIE}^{-1}[Y_d(k)] \quad (14)$$

DCT 영역에서의 이러한 처리 과정은 신호 $y(n)$으로부터 홀수 번째의 샘플만을 취함으로써, 결국 입력 신호를 새롭게 샘플링 하여, 우리가 원하는 샘플의 수를 가지는 신호를 얻을 수 있게 된다. 영상의 크기를 줄이는 리사이징의 경우에는 저주파 계수들을 취하고, 고주파 계수들은 버릴 것이기 때문에

$H^r(k) = \{1, 1, 1, 1, 0, 0, 0, 0\}$ 로 사용할 수 있다. $H^r(k)$을 역변환하면 다음과 같은 저주파 필터를 얻을 수 있다.

$$\begin{aligned} h^r(n) = \{&0.404517, 0.189883, \\ &-0.0384873, -0.0980449, \\ &-0.00792495, 0.0678165, \\ &0.030788, -0.0485469\} \end{aligned} \quad (15)$$

나. 영상의 크기를 늘이는 경우

영상의 크기를 늘이는 과정은 기본적으로 영상의 크기를 줄이는 과정과 마찬가지로 필터링 연산을 사용한다는 점에서 똑같다. 영상의 크기를 늘이기 위해서 가장 처음에 해야 하는 것은 DCT 영역에서 입력 신호를 업샘플링을 하는 것이다[10].

$$X_{uI}(k) = \frac{X_{II}(k) - X_{II}(N-k)}{\sqrt{2}} \quad (k = 0, 1, ..., N) \quad (16)$$

$X_{uI}(k)$는 $X_{II}(k)$가 업샘플링된 것이고, $X_{II}(k)$는 $x(n)$을 타입-2 DCT를 한 후 나머지 반에 '0'을 채워 넣은 것이다.

$$X_{II}(k) = \begin{cases} C_{IIE}\{x(n)\} & n, k = 0, 1, ..., N/2-1 \\ 0 & k = N/2, ..., N \end{cases} \quad (17)$$

(17) 식은 공간 영역에서 다음의 식과 일치한다.

$$x_u(n) = C_{IE}^{-1}[X_{uI}(k)] = \begin{cases} 0 & ,n \ even \\ x\left(\frac{n-1}{2}\right) & ,n \ odd \end{cases} \quad (18)$$

2. 나. 장에서도 언급했듯이, 입력 샘플들은 홀수 번째에 들어가게 되고 나머지 짝수 번째에는 '0'이 들어가게 된다. 이러한 과정이 끝난 후에는 저주파 필터링을 해야 한다. $x(0)$ 값과 $x(N)$ 값은 모두 '0' 이므로 가중치 함수는 사용되지 않는다. 다음의 필터링 과정은 영상의 크기를 줄이는 경우의 과정과 같다.

$$Y_{II}(k) = 2H^r(k)X_{uI}(k) \quad (19)$$

여기서 우리가 타입-2의 IDCT를 사용하여 역변환을 하면 결과는 다음과 같다.

$$y(n) = C_{IIE}^{-1}[Y_{II}(k)] \quad (20)$$

$$= 2h^r(n)\langle_{WSWS}SC_{WSWS}\rangle x_u(n) \quad (21)$$

다음 장에서 다룰 제안된 기본 알고리즘에서는, $H^r(k) = \{1, 1, 1, 1, 0, 0, 0, 0\}$를 사용하는 것과 마찬가지로 고주파 성분에 '0'을 넣는다. 요약하면, 제안한 알고리즘에서 채택된 방법은 $h^r(n)$을 공간영역에서 대칭 컨볼루션 하는 것과 일치한다. 각각의 경우에 사용되는 정확한 대칭 컨볼루션은 식 (13)과 식 (21)에 제시되어 있다.

3. DCT에 기반한 코덱을 고려한 변환 영역에서의 리사이징 알고리즘

가. 압축 효율을 개선하는 기본 알고리즘

그림 2는 제안하는 알고리즘을 개괄적으로 설명한 블록도이다. 영상은 8×8의 작은 영상들로 그룹화 되어 있고 각각의 작은 영상들은 변환 후에 양자화(quantiza-tion) 되어진다. 압축의 정도를 결정하는 유일한 변수

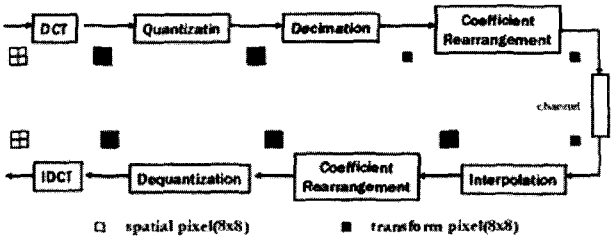


그림 2. 제안된 기본 알고리즘의 블록 다이어그램
Fig. 2. Block diagram of the proposed algorithm

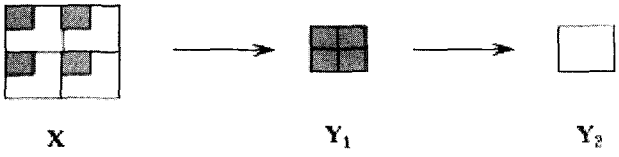


그림 3. DCT 영역에서 영상의 크기를 줄이는 과정
Fig. 3. Decimation in the DCT domain

는 화질 계수(quality factor, QF)이므로, 양자화표는 QF에 따라서 구성되어진다.

(1) 영상의 크기를 줄이는 경우

DCT 영역에서 크기를 줄이는 과정은 그림 3과 같다. 즉, 8×8의 작은 영상들은 4×4의 더 작은 영상으로 크기가 줄어들게 된다. 이러한 과정을 통해서 전체 영상의 크기가 수평 방향과 수직 방향으로 반씩 줄어들어 들게 되지만, 영상의 전체 크기가 줄었다고 해서 영상 데이터의 압축 효과는 발생하지는 않는다. 왜냐하면 버려지는 영역들은 런길이 코드(run length encoding, RLE)와 같은 방법을 사용해서 압축의 효율을 극대화할 수 있는 영역들이기 때문이다. 하지만 계수들을 재배열하게 되면 부가적인 압축 효과를 얻을 수 있다.

그림 3에서 X는 4개의 8×8 행렬로 이루어진 블록이다. 또한 F 행렬은 X를 Y_1으로 변환시키는 역할을 하며 다음과 같은 형태를 가진다.

$$F = \frac{1}{\sqrt{2}} \begin{bmatrix} I_4 & \Phi \end{bmatrix} \tag{22}$$

여기서 $1/\sqrt{2}$ 값은 전체 에너지를 보존하기 위해서 들어간 상수이다.

$$Y_1 = \begin{bmatrix} F & \Phi \\ \Phi & F \end{bmatrix} X \begin{bmatrix} F & \Phi \\ \Phi & F \end{bmatrix}^T \tag{23}$$

식 (23)에서 T는 전치(transpose)를 의미한다. 위의 변환을 통해서 원본 영상 X는 16×16 행렬에서 8×8 행렬로 그 크기가 줄어들고, 이러한 과정을 거치면 고주파 성분은 버려지게 된다.

하지만 앞에서도 언급했듯이, 위와 같은 과정으로 인해 그 크기가 줄어든 영상은 각각의 작은 영상들의 크기가 8×8에서 4×4로 줄었을 뿐 부가적인 압축 효과를 기대할 수는 없다. 만약 더 높은 압축 효율을 얻어야 한다면, Y_1 영상을 Y_2 영상으로 변환 시켜 주어야 한다. 이 과정은 위의 결과로 얻어진 Y_1의 4×4 크기의 4개의 작은 블록들을 각각 역변환을 하여 합쳐 8×8 영상을 만든 후에, 다시 8×8 변환을 해 주는 것이다.

$$Y_2 = C_8 \begin{bmatrix} C_4^{-1} & \Phi \\ \Phi & C_4^{-1} \end{bmatrix} Y_1 \begin{bmatrix} C_4^{-1} & \Phi \\ \Phi & C_4^{-1} \end{bmatrix}^T C_8^T \tag{24}$$

위의 식에서 C_8과 C_4^{-1}는 각각 8×8의 DCT 변환 행렬과, 4×4의 IDCT 행렬을 의미한다. Y_1에서 Y_2로 바꾸어주는 위의 과정으로 영상을 더 많이 압축할 수 있게 된다. 왜냐하면 저주파 성분만을 가진 4×4크기의 4개의 영상이 합쳐져서 저주파 성분과 고주파 성분이 모두 포함된 8×8 영상이 되기 때문에 고주파 성분을 양자화하여 더 높은 압축 효율을 낼 수 있기 때문이다. 다시 한 번 정리하면 식 (25)의 행렬을 가지고 있을 경우 DCT 영역에서 바로 영상의 크기를 줄이면서 동시에 더 큰 압축 효율을 얻을 수 있다.

$$T_{dec} = C_8 \begin{bmatrix} C_4^{-1} & \Phi \\ \Phi & C_4^{-1} \end{bmatrix} \begin{bmatrix} F & \Phi \\ \Phi & F \end{bmatrix} \tag{25}$$

X와 Y_2의 관계를 T_{dec}를 사용하여 나타내면 다음과 같다.

$$Y_2 = T_{dec} X T_{dec}^T \tag{26}$$

T_{dec}만 미리 저장해 놓는다면 실시간 구현이 가능하다. DCT 영역에서 블록간의 합침과 쪼갬은 [2]의 논문에 나와 있다. 이것의 fast 알고리즘은 [8]에서는 수학적인 방법에 의한 행렬 분리(separation)가 소개되어 있고, [11]에서는 C_8을 계산하기 위해서 C_4와 C_4^{-1}가 사용되어졌다.

(2) 영상의 크기를 늘이는 경우

영상의 크기를 늘이는 것은 영상의 크기를 줄이는 것의 역과정으로써 3. 가. (1)에 설명한 과정과 비슷한 과정을 거친다.

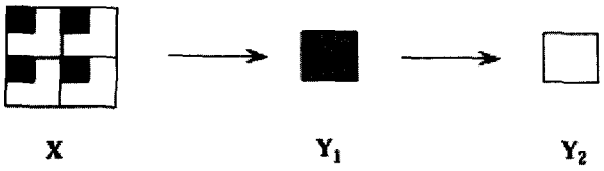


그림 4. DCT 영역에서 영상의 크기를 줄이는 과정
Fig. 4. Interpolation in the DCT domain

그림 4는 영상의 크기를 늘이는 과정을 보여준다. 행렬 G는 8×8 크기의 X를 고주파 부분에 '0'을 넣어 영상의 크기를 늘려 행렬 16×16 크기의 Y_1으로 만드는 역할을 하며, 다음과 같은 모양을 가지고 있다.

$$G = \sqrt{2}\begin{bmatrix} I_8 \\ \Phi_8 \end{bmatrix} \tag{27}$$

$\sqrt{2}$는 에너지를 유지하기 위해 쓰여졌다.

$$Y_1 = GXG^T \tag{28}$$

코덱의 요구조건을 맞추기 위해서는 16×16 크기의 Y_1을 8×8크기의 영상 4개가 합쳐진 Y_2로 재배열해야 한다.

$$Y_2 = \begin{bmatrix} C_8 & \Phi \\ \Phi & C_8 \end{bmatrix} C_{16}^{-1} Y_1 C_{16}^{-T} \begin{bmatrix} C_8 & \Phi \\ \Phi & C_8 \end{bmatrix}^T \tag{29}$$

여기서 식 (29)의 $-T$는 역변환의 전치를 의미한다. 만약 16×8 크기의 행렬 T_{int}를 식 (30)에서와 같이 정의 하고, 이 행렬을 안다고 가정하면, 공간 영역을 결치지 않고 DCT 영역에서 바로 영상의 크기를 늘릴 수 있다.

$$T_{int} = \begin{bmatrix} C_8 & \Phi \\ \Phi & C_8 \end{bmatrix} C_{16}^{-1} G \tag{30}$$

X와 Y_2의 관계를 T_{int}를 사용하여 다음과 같다.

$$Y_2 = T_{int}\, X\, T_{int}^T \tag{31}$$

나. 고주파 정보를 복구하는 확장 알고리즘

제안한 기본 알고리즘은 두 배의 압축 효율을 얻을 수 있지만 디코딩된 영상의 질은 기본 알고리즘을 적용하기 전의 영상만큼 좋지 않다. 특히 고주파가 많은 영상의 경우, 영상의 크기를 줄이기 전의 원래 영상보다 고주파 정보를 많이 잃게 되기 때문에 울림 현상(ringing artifact)이나 블록화 현상을 일으키게 된다. 이

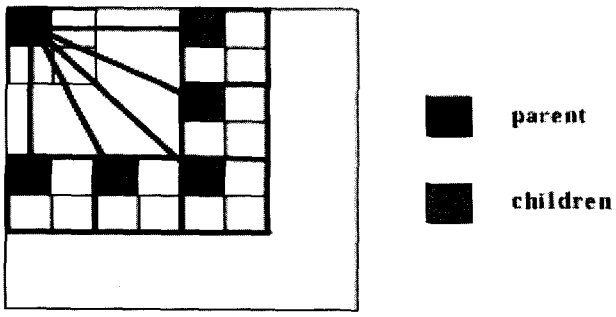


그림 5. 부모-자식 계수간의 관계
Fig. 5. Recovering the high-frequency content

와 같은 결과가 생기는 이유는 영상의 크기를 줄이는 과정에서 고주파 정보가 담겨있는 영역이 전혀 고려되지 않기 때문이므로, 가장 최소한의 비트를 더 추가하여 영상의 질을 개선해야 할 필요성이 생긴다.

그림 5에는 고주파 정보를 복원하기 위한 방법이 제시되어 있다. 위의 그림을 보면 기본 알고리즘에서는 전부 버려졌던 영역에 20개의 계수가 더해진 것을 볼 수 있다. 즉, 확장 알고리즘이란 고주파 성분이 상대적으로 많은 블록을 찾아 그 블록에 더 많은 비트를 할당하여 고주파 정보를 보존하는 알고리즘이다. 확장 알고리즘에서는 저주파의 4개의 계수만을 사용하여 부모-자식 관계를 이용하여 중간 주파수 부분의 20개의 계수만을 복원한다. 이 밖에 고주파의 나머지 계수들은 복원을 하더라도 영상의 질에 크게 영향을 끼치지 못함을 실험적으로 확인하였다. 자식에 해당하는 모든 계수들에는 각각 2비트가 할당되며, 2비트 중 1비트는 부호 비트 또 다른 1비트는 크기 비트로 사용하므로 2비트로 나타낼 수 있는 수는 -1, 0, 1의 3가지이다. 따라서 블록이 확장 알고리즘을 사용하면 기본 알고리즘을 통한 bit 수보다 총 4(부모 계수의 수)X5(각 부모 계수가 맡은 자식 계수의 수)X2(각 자식 계수를 표현하는데 드는 비트 수) = 40비트가 더해지게 된다. 확장 알고리즘에서 부모-자식 관계를 사용하는 이유는 인덱스를 코딩해야 하는 점을 없애기 위해서이다.

위에서 언급했듯이 확장 알고리즘이란 어떤 특정한 블록에만 적용되므로, 각 블록에 기본 알고리즘을 적용할 것인지 확장 알고리즘을 적용할 것인지 결정하는 일이 중요하다. 일반적으로 영상의 경계나 텍스쳐와 같은 고주파 정보가 많은 블록에 대해서는 확장 알고리즘을 사용하고, 물체의 내부나 배경과 같은 비교적 저주파 정보가 강한 블록에 대해서는 기본 알고리즘을 적용한다. 따라서 확장된 알고리즘과 기본 알고리즘을 적용하는 기준은 고주파 에너지이고, DCT 영역에서 고주파

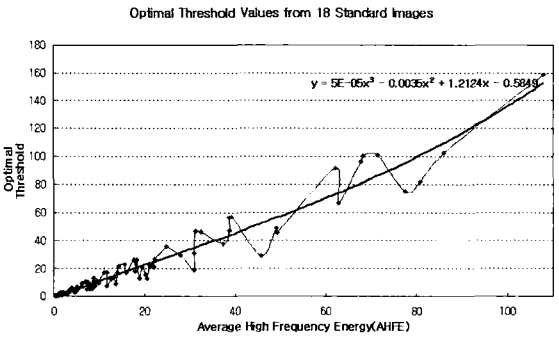


그림 6. 최적의 임계값을 구하는 실험
Fig. 6. Adaptive threshold values

에너지는 다음과 같이 구해진다.

$$HFE = \sum_{k_1=0}^{7}\sum_{k_2=0}^{7} X^2(k_1,k_2) - \sum_{k_1=0}^{3}\sum_{k_2=0}^{3} X^2(k_1,k_2) \tag{32}$$

HFE는 고주파 에너지(High Frequency Energy)를 의미한다. 이제 각 블록에 대해서 다음의 HFE를 계산해서 이 값이 임계값보다 크면 확장 알고리즘을 적용하고, 그렇지 않으면 기본 알고리즘을 적용하게 된다. 하지만, 임계값은 모든 영상에 대해서 고유하게 정해질 수밖에 없으므로 각 영상에 대해 임계값을 정하는 일이 중요하다. 임계값이 너무 크다면 적은 양의 블록들만이 확장 알고리즘에 적용되어 기본 알고리즘만을 적용한 결과와 차이가 없는 결과가 얻어질 것이고, 임계값이 너무 작을 경우에는 고주파 성분이 별로 없는 블록에 확장된 알고리즘이 적용되어 필요 없는 비트수가 증가하게 되는 문제점을 야기할 것이다. 따라서 임계값은 영상에 따라서 적절하게 고려되어야 한다. 기본적으로 임계값 t는 평균 고주파 에너지(average high- frequen -cy energy, AHFE)에 비례하고, QF에 반비례한다. 임계값 t와 AHFE, QF값의 관계는 다음과 같다.

$$t \propto \frac{AHFE}{QF} \tag{33}$$

$$AHFE = \sum_{i=0}^{N-1}\left\{\sum_{k_1=0}^{7}\sum_{k_2=0}^{7} X_i^2(k_1,k_2) - \sum_{k_1=0}^{3}\sum_{k_2=0}^{3} X_i^2(k_1,k_2)\right\}/N \tag{34}$$

N은 영상의 총 블록수를 의미한다. 하지만 QF는 AHFE에 영향을 미치기 때문에 단순히 임계값을 AHFE에 관한 함수로만 결정을 하면 식(35)와 같다.

$$t = f(AHFE) \tag{35}$$

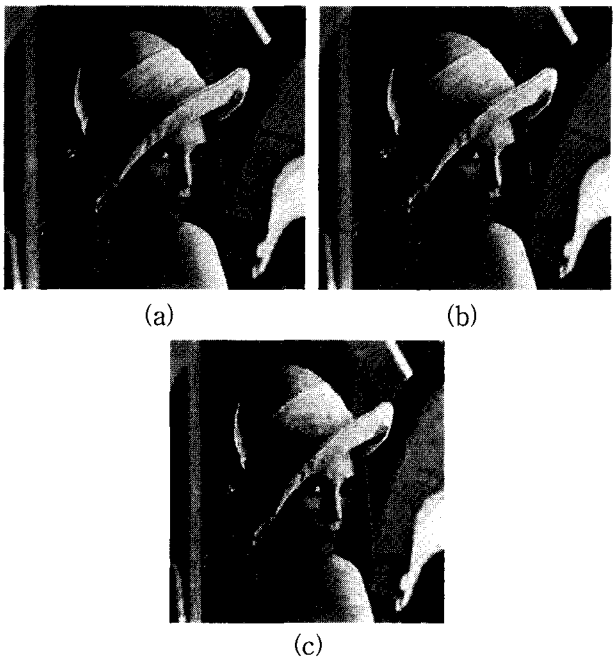


그림 7. QF 30으로 압축한 Lena 영상 (a) QF 30으로 압축한 JPEG 영상 (b) (a)에 기본 알고리즘을 적용했을 때 (c) (b)와 같은 크기의 JPEG 영상
Fig. 7. "Lena" compressed with QF=30; (a) JPEG image with QF=30; (b) image with the basic algorithm applied to (a), and (c) JPEG image of a size equal to that of (b)

최적의 $f(\cdot)$를 추정하기 위해서, 18장의 표준 영상을 실험에 사용하였다. 각각의 영상에서 AHFE를 계산하고 최적의 임계치를 찾아 실험 결과를 그림 6에 제시하였다. AHFE와 최적 임계치간의 관계를 다양한 차수의 다항식으로 추정해보면 다음과 같은 식이 얻어진다.

$$t = 1.2528x - 1.4021 \tag{36}$$

$$t = 1.25x^2 + 1.0029x + 0.2837 \tag{37}$$

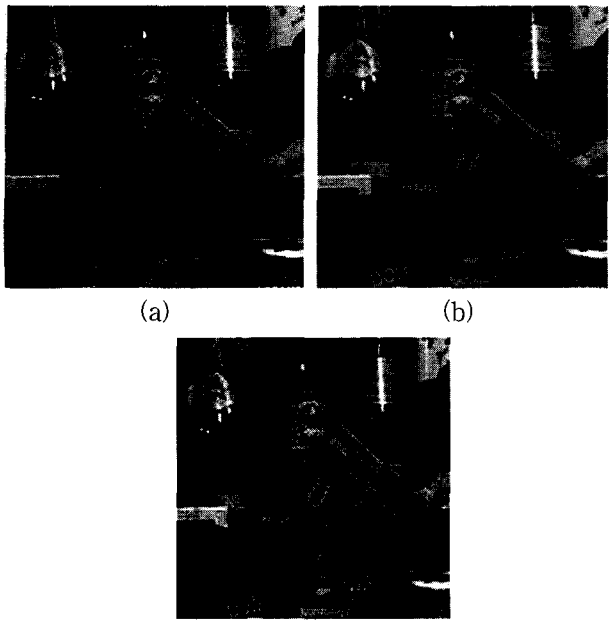
$$t = 5e^{-5}x^3 - 0.0035x^2 + 1.2124x - 0.5843 \tag{38}$$

식 (36), (37), (38)에서 x값은 AHFE를 나타낸다. 실험에서는 (38)식을 이용하였다.

III. 실 험

1. 기본 알고리즘

이 부분에서는 제안된 기본 알고리즘의 응용과 압축 효율의 개선 특성에 초점을 맞출 것이다. 실험에는 JPEG 표준에 기반한 영상을 사용하였다. 그림 7(a)는 QF가 30인 표준 JPEG 영상이고, 그림 7(b)는 제안한



(a) (b)

(c)

그림 8. QF 30으로 압축한 shopping 영상 (a) QF 30으로 압축한 JPEG 영상 (b) (a)에 기본 알고리즘을 적용했을 때 (c) (b)와 같은 크기의 JPEG 영상

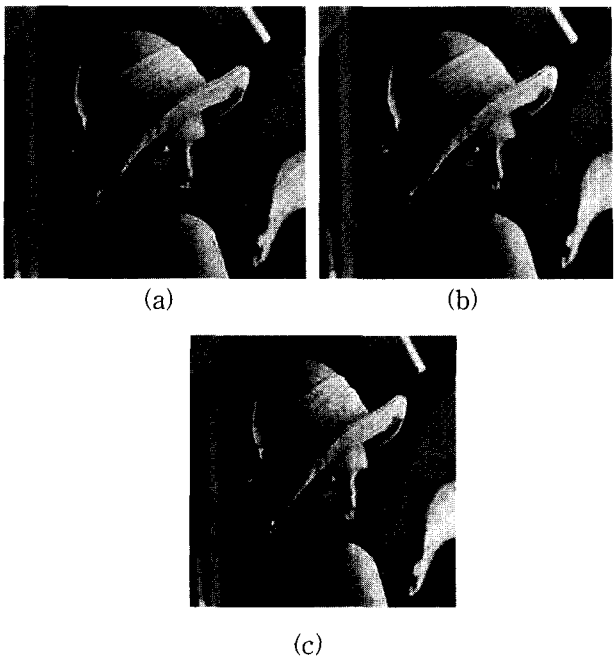
Fig. 8. "Shopping" compressed with QF=30; (a) JPEG image with QF=30; (b) image with the basic algorithm applied to (a), and (c) JPEG image of a size equal to that of (b)

표 1. 기본 알고리즘을 적용하기 전과 적용한 후의 영상의 PSNR 및 파일 크기 비교

Table 1. Objective quality and size before and after our basic algorithm is applied

Standard Images	Quality Factor	Before		After	
		SNR (dB)	Size (bytes)	SNR (dB)	Size (bytes)
"Lena"	30	17.43	4457	14.44	2259
"Shopping"	30	13.83	5250	10.08	2342

알고리즘을 적용한 결과이다. 그림 7(c)는 그림 7(b)와 동일한 크기를 가지는 QF가 8로 압축된 JPEG 영상이다. 그림 7(b)와 그림 7(c)는 같은 비트수를 가지지만 두 영상의 질에는 상당한 차이가 있다. 그림 7(c)는 많은 블록킹 효과가 나타나고 있고 에지를 따라서 울림 현상도 발생하고 있다. 따라서 기본 알고리즘을 사용한 그림 7(b) 영상은 그림 7(c)의 JPEG 영상보다 질적으로도 양적으로도 뛰어난 영상이다. 그림 8을 보면 기본 알고리즘을 적용하였을 때의 부수적인 효과를 확인할



(a) (b)

(c)

그림 9. QF 30으로 압축한 Lena 영상 (a) QF 30일 때의 기본알고리즘 적용 영상 (b) (a)에 확장 알고리즘을 적용한 영상 (c) (b)와 같은 크기의 JPEG 영상

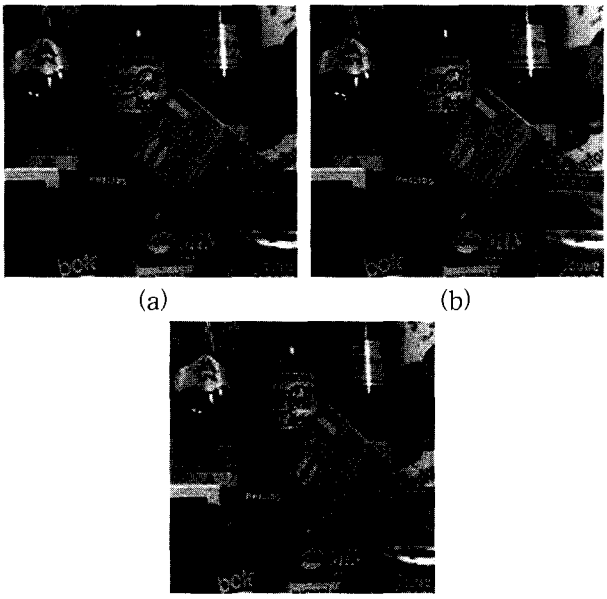
Fig. 9. "Lena" compressed with QF=30; (a) image with the basic algorithm with QF=30 applied, (b) image with the extended algorithm applied, to (a), and (c) JPEG image of a size equal to that of (b)

수 있다. 각각의 8×8블록이 대응하는 16×16 블록의 저주파 영역으로 옮겨지면서 16×16안에 블러(blur) 현

상이 발생하게 되고 그 결과로 16×16블록 안에 있는 8×8 블록들끼리는 블록킹 효과가 발생하지 않게 되어 전체적으로 블록킹 효과가 감소하는 효과를 얻을 수 있다. 그림 8은 고주파 성분이 많은 "shopping" 영상을 가지고 실험한 결과이다. 이런 고주파 성분이 많은 영상의 경우 알고리즘 과정에서 버려지게 되는 고주파 성분이 많기 때문에 압축 효율이 증가하는 대신에 영상은 열화 정도가 커지게 된다. 따라서 기본 알고리즘은 평평한 영상이나, 저비트율 환경에서 압축 된 영상에 효율적이라고 할 수 있다. 표 1에는 기본 알고리즘을 적용하기 전과 적용한 후의 영상의 PSNR과 파일 크기가 구체적으로 제시되어 있다.

2. 확장 알고리즘

이 장에서는 기본 알고리즘을 적용했을 때와 비교하여 확장 알고리즘을 사용할 시, 기본 알고리즘과 비교



(a) (b)

(c)

그림 10. QF 30으로 압축한 shopping 영상 (a) QF 30일 때의 기본 알고리즘 적용 영상 (b) (a)에 확장 알고리즘을 적용한 영상 (c) (b)와 같은 크기의 JPEG 영상

Fig. 10. "Shopping" compressed with QF=30; (a) image with the basic algorithm with QF=30 applied, (b) image with the extended algorithm applied, to (a), and (c) JPEG image of a size equal to that of (b)

표 2. 확장 알고리즘을 적용하기 전과 적용한 후의 영상의 PSNR 및 파일 크기 비교

Table 2. Objective quality and size before and after our extended algorithm is applied

Standard Images	Quality Factor	Before		After	
		SNR (dB)	Size (bytes)	SNR (dB)	Size (bytes)
"Lena"	30	14.44	2259	16.23	2422
"Shopping"	30	10.08	2342	11.62	2636

하여 고주파 정보가 얼마나 많이 복원 되어지는가에 초점을 맞추겠다. 그림 9는 QF가 30으로 압축된 "Lena" 실험 영상을 사용하였고, 그림 10에서는 QF가 마찬가지로 30으로 압축된 "shopping" 실험 영상을 사용하였다. 그리고 확장된 알고리즘을 적용할 것인지 기본 알고리즘을 적용할 것인지 결정하는 임계값은 두 가지 영상에 대하여 식 (38)을 똑같이 적용하였다. 첫 번째 열은 기본 알고리즘을 적용한 영상들이고, 두 번째 열은 확장 알고리즘을 적용한 영상들이다. 마지막으로 세 번째 열은 두 번째 열과 같은 비트율을 가지도록 압축된 JPEG 표준 영상이다. 2가지 모든 경우에서, 기본 알고리즘보다 200 바이트정도를 더 할당한 제안한 확장 알고리즘의 경우 2~3dB정도의 SNR 개선을 보였다. 확장 알고리즘은 "Lena"와 같은 비교적 평탄한 영상이나 높은 비율로 압축된 영상에서는 특히 획기적인 영상의 질적 개선을 이룰 수 있었다. 이러한 사실을 바탕으로 제안한 알고리즘이 평탄한 영상이나 저비트율로 압축된 영상에 아주 적절한 알고리즘임을 다시 확인 수 있다. 표 2에 상세한 실험 결과를 제시하였다.

Ⅳ. 결 론

지금까지 기본 알고리즘과 확장된 알고리즘의 유용성과 그 응용성에 대해서 설명하였다. 기본 알고리즘을 사용함으로써 부가적인 압축 효율을 얻을 수 있으며, 확장된 알고리즘은 기본 알고리즘에서 고려하지 않았던 고주파 정보를 고려하여 기본 알고리즘에서 발생하는 영상의 열화를 막을 수 있음을 확인하였다. 제안된 알고리즘은 전체 이미지에서 블록화 현상을 일으키게 되는 공간 영역에서의 리사이징 알고리즘을 거치지 않고 변환 영역에서 독립적으로 수행되어지기 때문에 제안된 알고리즘은 시각적으로 더 좋은 영상을 얻어낼 수 있으며, Ⅰ장에서 언급되었던 리사이징을 하는데 있어서의 몇 가지 어려움을 극복해 내었다. 실험 결과, 제안된 알고리즘은 저비트율 환경에서 압축된 영상의 경우에 좋은 성능을 가지고 있기 때문에 적은 대역폭이 요구되는 셀룰러 네트워크, 화상 회의와 같은 분야에 응용되어질 수 있다.

참 고 문 헌

[1] Haiyan Shu and Lap-Pui Chau, "A Fast Arbitrary Downsizing Algorithm for Video Transcoding", IEEE ICIP 2003, Barcelona, Spain, Sep. 2003

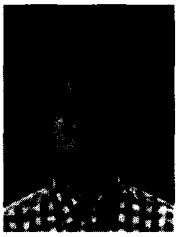
[2] D. Mehta and U. B. Desai, "A Primer To Video Transcoding : Image Transcoding", Proceedings, Eighth National Conference on Computing, Jan. 2002

[3] Qingwen Hu, Sethuraman Panchanathan, "Image/Video Spatial Scalability in Compressed Domain", IEEE Transactions on Industrial Electronics, Vol. 45, No. 1, Feb. 1998.

[4] Jae Beom Lee, Byeong Gi Lee, "Transform

Domain Filtering Based on Pipelining Structure'', IEEE Transactions on Signal Processing, Vol. 40, No. 8, Aug. 1992.
[5] Neri Merhav, Vasudev Bhasharan, "Fast Algorithms for DCT-Domain Image Down-Sampling and for Inverse Motion Compensation", IEEE Transactions on Circuits and Systems for Video Technology, Vol. 7, No. 3, Jun. 1997.
[6] Robert G. Keys, "Cubic Convolution Interpolation for Digital Image Processing'', IEEE Transactions on Acoustics, Speech, and Signal Processing, Vol. ASSP-29, No. 6, Dec. 1981.
[7] Stephen A. Martucci, "Symmetric Convolution and the Discrete Sine and Cosine Transforms'', IEEE Transactions on Signal Processing, Vol. 42, No. 5, May. 1994.
[8] Rakesh Dugad, Narendra Ahuja, "A Fast Scheme for Image Size Change in the Compressed Domain'', IEEE Transactions on Circuits and Systems for Video Technology, Apr. 2001.
[9] Discrete Cosine Transform, K.R.Rao, P.Yip, Academic Press, 1990.
[10] Stephen A. Martucci, "Image Resizing in the Discrete Cosine Transform Domain", ICIP 1995, Vol. 2, May. 1995.
[11] Skodras A. N., Christopoulos C. A., "Down-Sampling of Compressed Images in the DCT Domain", VIII European Signal Processing Conference, EUSIPCO-98, Island of Rhodes, Greece, pp. 8-11, Sep. 1998.

저 자 소 개



신 건 식(학생회원)
2000년 연세대학교 기계전자 공학부 공학부 학사 졸업
2002년 연세대학교 전기전자 공학과 석사 졸업
현재 연세대학교 전기전자공학과 박사 과정
<주관심분야: 영상 리사이징, 변환영역에서의 프로세싱, 웨이블릿을 이용한 잡음제거>

장 준 영(학생회원)
2004년 연세대학교 전기전자공학부 학사 졸업
현재 연세대학교 전기전자공학과 석사 과정
<주관심분야: 공간 및 변환 영역에서의 잡음 제거>

강 문 기(정회원)
1986년 서울대학교 전자공학과 학사 졸업
1988년 서울대학교 전자공학과 석사 졸업
1994년 Northwestern University 박사 졸업
1994년~1997년 Assistatnt Professor (University of Minnesota, Duluth)
1997년~1999년 연세대학교 전기전자공학과 조교수
1999년~2004년 연세대학교 전기전자공학과 부교수
현재 연세대학교 전기전자공학과 교수
<주관심분야: 영상복원, 초해상도 영상복원, 비선형 필터링, 비디오 분석 및 처리 부화소단 위의 움직임 추정>