DCT 맵 FSVQ와 단방향 분포 허프만 트리를 이용한 영상 압축

조 성 환†

요 약

본 논문에서는 영상 전송을 위한 벡터 양자화기를 설계할 때 2차원 DCT에 근거한 DCT 맵과 유한상태 벡터 양자화를 이용하는 새로운 부호책(codebook) 설계 알고리듬을 제안한다. 영상을 윤곽선이 많은 부분과 적은 부분으로 나누어 맵을 만들고 이 맵에 따라 영상의 중요한 특징들을 2차원 DCT로 추출한다. 유한상태 벡터 양자화기의 마스터 부호책은 트리 구조에 근거한 2진 트리를 사용하여 두 영역을 따로 학습세트로 나눔으로서 만들어진다. 이와 같이 작성된 마스터 부호책으로부터 상태 부호책을 작성하여 입력 벡터에 대하여 마스터 부호책이 아닌 상태 부호책으로부터 부호단어를 찾는다. 또한 인덱스의 부호화는 고속 디지털 전송에 중요한 부분이기 때문에 고정길이의 부호를 엔트로피 부호화 법칙에 따라 가변 길이의 부호로 바꾸어 수행한다. 즉, 설계한 부호책에서 각 부호에 전송 부호 할당은 허프만 부호화를 수행하는데, 허프만 트리에서의 허프만 코드의 생성을 빠르게 하기 위해 본 논문에서는 트리의 단방향 분포 허프만 트리 알고리듬을 제안한다. Einstein과 Bridge 영상에 대하여 본 알고리듬으로 영상을 부호화했을 때 PNN 알고리듬보다는 각각 2.04 dB과 2.48 dB만큼, CVQ 알고리듬보다 각각 약 1.75 dB과 0.99 dB만큼 더 좋은 영상의 화질을 얻을 수 있었다.

Image Compression Using DCT Map FSVQ and Single-side Distribution Huffman Tree

Seong-Hwan Cho†

ABSTRACT

In this paper, a new codebook design algorithm is proposed. It uses a DCT map based on two-dimensional discrete cosine transform (2D DCT) and finite state vector quantizer(FSVQ) when the vector quantizer is designed for image transmission. We make the map by dividing input image according to edge quantity, then by the map, the significant features of training image are extracted by using the 2D DCT. A master codebook of FSVQ is generated by partitioning the training set using binary tree based on tree-structure. The state codebook is constructed from the master codebook, and then the index of input image is searched at not master codebook but state codebook. And, because the coding of index is important part for high speed digital transmission, it converts fixed length codes to variable length codes in terms of entropy coding rule. The huffman coding assigns transmission codes to codes of codebook. This paper proposes single-side growing huffman tree to speed up huffman code generation process of huffman tree. Compared with the pairwise nearest neighbor (PNN) and

† 정 회 원:대유공업전문대학 전자계산기과
논문접수:1997년 3월 7일, 심사완료:1997년 8월 29일

classified VQ (CVQ) algorithm, about Einstein and Bridge image, the new algorithm shows better picture quality with 2.04 dB and 2.48 dB differences as to PNN, 1.75 dB and 0.99 dB differences as to CVQ respectively.

1. 서 론

최근 몇년동안 벡터 양자화는 영상이나 음성의 효과적인 데이터 압축기술로 알려져 왔고 또 이에 대한 여러 가지 논문이 발표되었다[1-3]. Gersho와 Gray는 벡터 양자화에 대한 부호화 구조와 디자인 알고리듬, 그리고 응용의 여러 분야와 함께 벡터 양자화의 기본 원리 개발을 설명하였다[4]. 간단한 벡터 양자화는 k 차원의 유클리디언 공간 R^k의 Q를 R^k의 유한 부분 집합 Y로의 맵핑으로 정의할 수 있다. 즉 $Q: R^k \rightarrow Y$로 쓸 수 있는데 $Y = \{y_i; i = 1, 2, ..., N\}$는 재구성 벡터 집합이고 N은 Y에서 벡터의 수이다. 이때 부분 집합 Y를 부호책(codebook)이라 부른다. 즉, 일반적인 벡터 양자화기에서는 입력을 부호책의 일정수의 벡터 집합에 속하게 하여 복호화기로는 그 집합의 인덱스만을 보내 복호화기에서는 부호화기와 같은 부호책을 이용하여 간단한 테이블 참조(table look-up)방법을 사용하여 원 벡터를 재생한다. 벡터 양자화의 관건은 벡터를 나타내는 좋은 부호책을 설계하는 것인데, 잘 알려진 부호책 설계는 Lloyd 방법[5]과 K-means clustering 방법[6]을 이용한 Linde, Buzo, Gray의 방법(LBG 알고리듬)[7]이다. 이 알고리듬에서는 모든 학습 벡터가 부호벡터와의 최소 왜곡 법칙에 따라 클러스터를 이룬다. 이들 클러스터의 중심값(centroid)이 다음 반복에서 새로운 부호 벡터가 되는데, 이와 같은 단계를 부호벡터와 클러스터 구성원사이의 전체 왜곡이 정해진 기준 이하가 될 때까지 계속한다. 그러나 LBG 알고리듬은 부호책 설계를 시작하기 위해서는 초기 부호책이 제공되어야 하고 그 단계가 반복적이며, 각 부호 벡터와 클러스터 구성원사이에 왜곡(distortion)계산으로 인한 많은 수렴시간을 요구한다.

또한 Equitz에 의해 PNN(pairwise nearest neighbor)이라는 알고리듬이 제안되었는데[8, 9], 이 알고리듬의 기본 개념은 두 클러스터의 중심값에 의해 두개의 가장 가까운 클러스터를 다른 클러스터로 연속적으로 합병하는 것이다. 즉 전체 학습 집합을 가지고 시작하여 벡터 집합을 원하는 크기로 줄이거나 부호 왜곡에 도달할 때까지 단계를 반복한다. PNN 알고리듬은 LBG 알고리듬과 달리 초기 부호책이 필요 없고, 실행시간은 LBG 알고리듬 실행 시간의 약 5% 정도이다[8].

벡터 양자화기의 한 예로써 영상의 시각적인 특징(perceptual feature)을 유지하기 위해서 분류 벡터 양자화(Classified VQ:CVQ)가 Ramamurthi와 Gersho에 의해 제안되었다[10]. 이 CVQ에서는 윤곽선(edge)의 방향, 위치 등에 의해서나 배경의 밝기 영역에 따라 각 블록을 분류하고, 준 부호책(subcodebook)이라고 하는 분리된 부호책이 각각의 분류에 대해 속하는 학습 벡터로부터 만들어지는데, 이 각 부호책의 부호벡터는 LBG 알고리듬을 사용하여 할당한다. 처음 영상의 입력벡터가 부호기로 들어오면 분류기는 그 블록을 각 특징 클래스중의 하나로 분류하여 그 클래스의 부호단어를 전송하고, 복호기는 전송된 부호단어로부터 전송된 부호책을 이용하여 블록에 해당되는 벡터를 만들어준다. 이 CVQ는 각 준 부호책에 반복적이고 초기값이 요구되는 LBG 알고리듬을 사용하므로 알고리듬 수행 시간이 상당히 길다.

이와 같은 일반적인 벡터 양자화 알고리듬들은 이웃 화소 사이의 높은 상관관계를 이용하지만 이웃 블록간의 상관관계는 무시한다. 이에 대한 대안으로서 일반적인 벡터 양자화기로 가능한 것 이하로 비트율을 감소시키기 위해 블록간 상관관계를 이용하는 유한상태 벡터 양자화기(FSVQ)를 사용한다. 유한상태 벡터 양자화기는 부호화기 상태에 따라서 자신의 부호책을 가지는 벡터 양자화기로써 현재 입력 벡터의 상태는 이전에 부호화된 벡터에 의해 정의된다. 각각의 입력 벡터는 현재 부호화기 상태에 의하여 결정된 부호책을 사용하여 부호화된다.

본 논문에서는 분류화 문제에 있어서 영상의 주파수 성분에 따라 DCT 맵을 작성하여, 각 맵 값에 따라 데이터의 차원(dimension)을 줄이는 특징의 추출방법과, 분류기를 설계할 때 특징은 2차원 DCT의 계수값을 이용하여 원 영상으로부터 추출한다. 각 부호책의 부호단어(codeword)를 작성하기 위해 이진(binary)

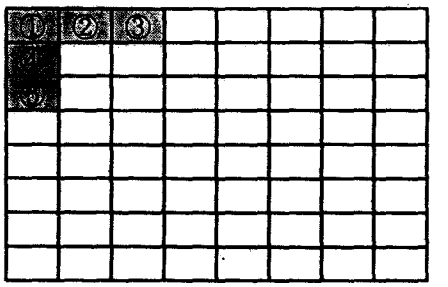
트리 분류기를 사용한다. 본 논문의 알고리듬은 초기 부호책이 필요 없고, 영상의 각 블록에 대한 부호책으로부터의 부호단어 선택은 유한상태 벡터 양자화기를 이용하며, 부호책의 각 부호단어에 대한 인덱스를 전송할 때에 그 인덱스는 허프만(Huffman) 부호화에 따라 부호화한다. 허프만 부호화는 가장 간단한 형태로서 이진 트리로 나타낼 수 있다[11, 12]. 그러나 가변 길이 부호화이기 때문에 허프만 트리는 트리의 뿌리로부터 대칭이 아닌 한쪽 방향으로 기울어지는 트리가 만들어진다. 트리구조에서 이와 같은 불균형성은 심볼을 위치시키기 위한 긴 검색과정의 결과이기도 하다. 본 논문에서는 심볼에 대한 고속 검색을 수행하기 위해 부호 길이에 근거한 가변 길이 부호를 배열시키는 알고리듬을 제안한다. 자주 사용되는 심볼은 비교적 짧은 코드를 할당하므로써 덜 사용되는 심볼에 대해 검색시간을 줄일 수 있어서 전체 검색 지연 시간을 감소시킬 수 있다. 시뮬레이션 결과 본 논문에서 제안한 방법으로 영상 부호화를 했을 경우 PNN이나 CVQ 알고리듬과 비교하여 계산 시간을 줄이고 더 좋은 화질의 영상을 얻을 수 있었다.

2. DCT 맵 작성

부호책을 설계할 때, 영상의 모든 블록들에 대해 같은 크기의 부호책을 설계하는 것보다는 부호화하기 쉬운 부분, 즉 윤곽선(edge)을 적게 포함하고 있는 비윤곽선(non-edge) 블록은 적은 수의 부호책을, 윤곽선 즉 고주파 성분이 많은 블록은 더 큰 수의 부호책을 설계하면 같은 평균 제곱 오차를 갖는 복원 영상에 대해 압축율을 높일 수 있다. 따라서 본 논문은 영상의 각 블록에 대해 고저주파의 포함 여부를 결정하기 위하여 2차원 DCT를 계산하여 DCT 맵을 작성한다.

DCT 맵은 다음과 같은 세 단계로 작성한다.

[단계 1] 원 영상을 좌측상단으로부터 우측으로 8 화소, 하단으로 8 화소씩 8×8의 부블록(subblock)으로 나누어 2차원 DCT를 계산한다. 계산 결과에 대해 (그림 1)과 같이 좌측 상단의 그 블록의 DC 평균값인 ①의 값과 낮은 주파수의 AC 값인 ②, ③, ④, ⑤의 값을 포함한 5개 항의 값은 그대로 유지하고 나머지 값은 0으로 만들어 2차원 DCT의 역변환을 취한다. 그 결과 값들과 원 영상의 블록값들과의 평균 제곱 오차를 계산하여 그 값이 임계값(threshold) δ보다 크면 이 블록에 대한 DCT 맵은 "1"로, 적으면 "0"으로 맵의 값을 결정한다.



(그림 1) 8×8 블록에서의 DCT값 선택
(Fig. 1) The choice of DCT values in 8×8 subblock

[단계 2] DCT 맵에서 "1"로 된 각각의 8×8 블록에 대해 4개의 4×4 블록으로 나누어 단계 1과 마찬가지 방법으로 4개의 4×4 블록에 대해 DCT 맵의 값을 결정한다. "0"으로 된 8×8 블록에 대해서는 4개의 4×4 블록에 대한 DCT 맵 값을 "0"으로 한다.

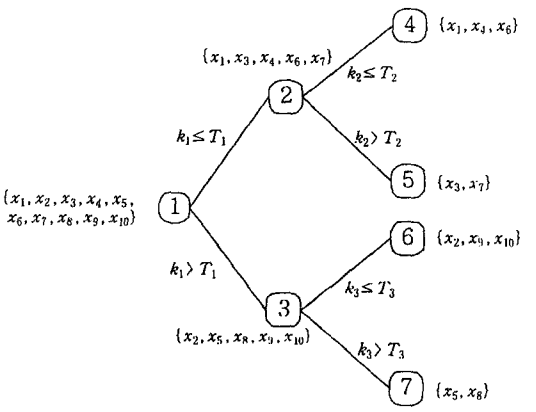
[단계 3] DCT 맵이 "1"로 정해진 4×4 블록과 "0"인 4×4 블록들을 따로 모아 두개의 다른 부호책 크기를 갖는 2진 트리 구조 양자화기의 입력으로 들어간다.

3. 이진 트리 분류기를 이용한 부호책 설계

이진 트리는 가장 간단하고 널리 사용[13-15]되는 결정 트리 구조로서 하나의 뿌리(root) 노드와 일련의 단말, 비단말 노드로 구성되며, 비단말 노드는 2개의 하위 노드를 가진다. 패턴 벡터를 뿌리 벡터로부터 각 단말까지 트리를 통하여 이동함으로써 분류한다. 각 비단말 노드는 그 하위노드 중 하나에 입력 패턴을 할당하기 위해 특징 집단을 사용하고 단말노드는 분류된 패턴들에 할당되는 대표 패턴(일반적으로 그 노드의 중심값)을 가진다.

본 논문에서는 벡터 양자화기 부호책을 만들기 위해 이진 트리 분류기를 사용하는데 학습 벡터로부터 축출된 특징들을 임계값(threshold)으로 간단히 분류한다. (그림 2)에 부호책 설계의 예를 나타내었는데,

(그림 2)에서 학습 벡터 $\{x_1, x_2,..., x_{10}\}$로부터 4개 크기의 부호책을 만든다고 하면, 뿌리(①)노드에서 이 10개의 학습벡터를 두개의 하위(②, ③)노드로 분류하는데, 각 벡터의 키 값(k_1)이 임계값(T_1)보다 작거나 같으면 ②번 노드로, 크면 ③번 노드로 각 벡터를 분류한다. 그 다음 ②번 노드에서도 학습벡터 $\{x_1, x_3, x_4, x_6, x_7\}$을 키 값($k_2$)과 임계값($T_2$)을 비교하여 작거나 같으면 ④번 노드, 크면 ⑤번 노드로 분류하고 ③번 노드도 마찬가지로 수행한다. 이렇게 해서 만들어진 단말노드(④, ⑤, ⑥, ⑦)의 분류된 벡터들의 중심값(centroid)을 계산하여 그 값을 부호책의 부호 벡터로 사용한다. 예를 들어, ⑥번 노드의 부호단어 벡터 C_6은 $(1/3)\{x_2 + x_9 + x_{10}\}$과 같이 계산한다.



(그림 2) 이진 트리 분류기를 이용한 부호책 설계
(Fig. 2) The Codebook design using binary tree classifier

(그림 2)의 이진 트리 분류기에서 "키" 값의 선택을 위해 영상의 각 블록에 DCT를 계산한다. 이 DCT는 실(real)함수 계산이고, 계산의 복잡도가 비교적 낮으며, 고속 알고리듬[16-18]이 존재하기 때문에 많은 알고리듬 수행시간을 필요로 하지 않고, 또 이진 트리 분류기의 부호책 설계 전에 이미 DCT 맵에서 그 블록에 대한 DCT 값을 계산하였기 때문에 그 값을 이용하면 더욱 알고리듬 수행 시간을 줄일 수 있다.

원 영상을 M×M 크기의 겹치지 않는 블록으로 나누어, 각 블록에 대해 이차원 DCT를 취한다[19]. 계산된 이차원 DCT 계수들은 (그림 3)과 같이 정지영상 표준안인 JPEG에서 사용하는 것처럼 지그재그 스캔으로 1차원으로 재배열하게 되는데 이는 DCT 계산 후의 그 계수 값들을 DC 값, 저주파수 값으로부터 고주파수 값 순으로 재배열시키기 위해서이다. (그림 3)은 4×4 블록의 예인데 블록내의 번호 순서에 따라 재배열된 1차원의 계수 값들을 그 블록의 특징벡터로 한다. 이와 같은 1차원 배열의 특징 벡터를 이진 분류 트리의 분류기로 입력한다.

행 \ 열	①	②	③	④
①	■	2	6	7
②	3	5	8	13
③	4	9	12	14
④	10	11	15	16

(4×4 DCT 계수 블록인 경우이고 ■는 DC 계수 값)

(그림 3) DCT 계수의 지그재그 스캔
(Fig. 3) The zigzag scan of the DCT coefficients

이진 트리 분류기로 입력되는 각 특징 중에서 각 노드에서 분류를 위한 키의 선택은 가장 큰 분산값을 갖는 특징을 키로 한다. 이는 데이터가 특정된 특징에 대해 넓게 퍼져 분포되어 있다면 그 특징에서의 차이가 가장 중요한 점이 되기 때문이다. 그리고 각 노드의 임계값으로는 선택된 키에 해당하는 특징에 대한 평균값을 사용하는데 이는 평균값이 특징의 대부분의 통계적 정보를 포함할 뿐만 아니라 계산이 간단하기 때문이다.

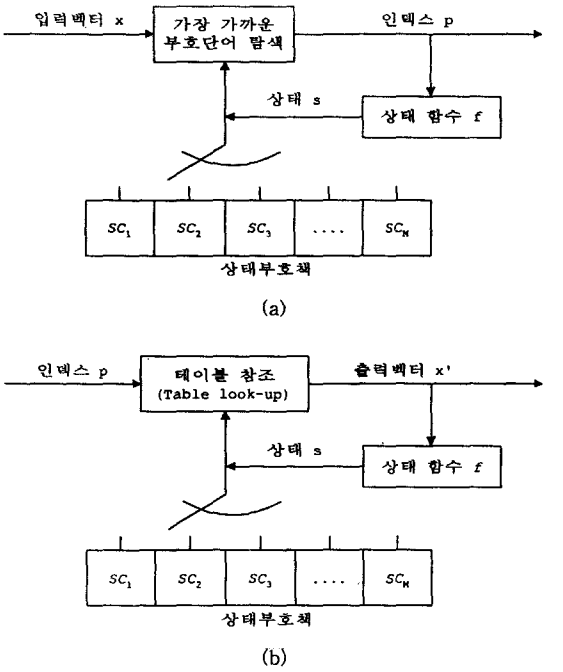
예를 들어 (그림 2)에서 설명한 트리 분류기에서 ②번 노드의 학습벡터를 1차원으로 나타낸 것을 D라 할 때 이 값들이 〈표 1〉에 나타나 있다. 〈표 1〉에서와 같이 각 벡터에서 각 특징에 대한 평균값 $\{M(1),..., M(16)\}$과 분산값 $\{V(1),..., V(16)\}$을 계산한 후 가장 큰 분산값을 갖는 특징이 키로서 선택되고 그 키에 해당되는 평균값을 임계값으로 한다. 예로서 $V(1)$ 값이 가장 크다면 이 ②번 노드의 임계값은 $M(1)$이 되

는 것이다. 그래서 각 벡터의 첫 번째 특징 벡터값, 즉 $D_1(1)$, $D_3(1)$, $D_7(1)$, $D_{11}(1)$과 $M(1)$을 비교하여 작거나 같으면 그 벡터는 ④번 노드로, 크면 ⑤번 노드로 분류한다. 이진 트리의 단말 노드의 수, 즉 부호책의 부호 단어의 수가 원하는 크기가 될 때까지 이 방법을 계속 실행하여 부호책의 부호 단어의 수가 이진 트리의 단말 노드의 수와 같게 되면 마지막으로 각 단말 노드에서의 벡터들의 중심값을 계산하여 부호책의 부호 벡터로 사용한다. 단말 노드의 중심값은 그 노드에 속해 있는 벡터들의 각 특징들에 대한 평균값으로 한다.

〈표 1〉 노드의 키 선택

〈Table 1〉 The choice of key at node

학습벡터	x_1	x_3	x_4	x_6	x_7	평균	분산
계수벡터	D_1	D_3	D_4	D_6	D_7		
DCT 계수값	$D_1(1)$	$D_3(1)$	$D_4(1)$	$D_6(1)$	$D_7(1)$	$M(1)$	$V(1)$
	$D_1(2)$	$D_3(2)$	$D_4(2)$	$D_6(2)$	$D_7(2)$	$M(2)$	$M(2)$
	⋮	⋮	⋮	⋮	⋮	⋮	⋮
	$D_1(16)$	$D_3(16)$	$D_4(16)$	$D_6(16)$	$D_7(16)$	$M(16)$	$V(16)$



(그림 4) 유한상태 벡터 양자화기. (a) 부호화기 (b) 복호화기

(Fig. 4) Finite State Vector Quantizer. (a) Encoder (b) Decoder

4. 유한상태 벡터 양자화기(FSVQ)

영상 부호화에서 이웃 블록들 사이의 상관관계를 이용하면 복원 영상의 화질을 개선하고 벡터 양자화기의 전송 비트율을 감소시킬 수 있다. 유한상태 벡터 양자화기(Finite State Vector Quantizer : FSVQ)는 영상의 이웃 블록간의 높은 상관관계를 이용하여 이전 블록의 상태를 기억하여 부호화를 수행한다[20-24].

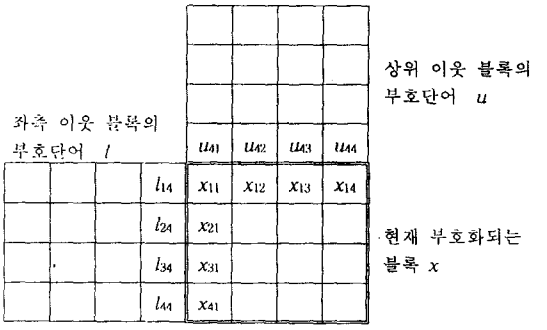
유한상태 벡터 양자화기는 일반적인 VQ의 유한한 집합, 즉 부호화기 상태에 따라서 자신의 부호책을 가지는 벡터 양자화기로 생각할 수 있다. 현재 입력 벡터의 상태는 이전에 부호화된 벡터에 의해 정의되며, 각각의 입력 벡터를 현재 부호화기 상태에 의하여 결정된 부호책을 사용하여 부호화한다. (그림 4)에는 유한상태 벡터 양자화를 수행하는 부호화기와 복호화기가 나타나 있다.

여기서 $S=\{s_i : i=1,...,M\}$은 상태 공간이다. 각 상태 s_i에 대하여 s_i의 상태 부호책(State codebook)으로서 마스터 부호책(Master codebook) $Y=\{y_i : i=1,...,N_m\}$로부터 N_f개의 부호단어를 선택한다. 이전에 부호화된 벡터의 부호단어를 p라 할 때, 상태 함수 $f(p)$가 현재 입력 벡터 x의 상태를 결정하기 위하여 사용된다. 입력 벡터 x를 부호화하기 위하여 부호화기는 현재 상태 $s=f(p)$를 찾고 그에 해당하는 부호단어를 찾기 위하여 마스터 부호책 Y가 아닌 상태 부호책 SC_s를 탐색한다. 일반적으로 상태 부호책의 부호단어 수인 N_f는 마스터 부호책의 부호단어 수 N_m보다 훨씬 적다. 현재 부호화하려는 블록에 대해 상태 부호책을 선택하고, 그 상태 부호책으로부터 가장 작은 왜곡을 가지는 부호 단어를 선택하여 그 인덱스를 복호화기로 전송한다.

5. 유한상태 벡터 양자화기를 이용한 영상의 부호화

유한상태 벡터 양자화기에서는 마스터 부호책으로부터 상태 부호책을 설계하는 알고리듬이 있어야 한다. 우선 마스터 부호책의 설계가 필요한데 일반적으로 많이 사용하는 것이 LBG 알고리듬이다[24]. 그러나 LBG 알고리듬은 초기 부호책이 필요하고 알고리듬 실행시간이 긴 단점으로 인하여 본 논문에서는 2절과 3절에서 제안한 DCT 맵과 이진 트리 분류기를 이용한 알고리듬으로 마스터 부호책을 작성한다. 즉, DCT 맵에 의해 각각 1과 0으로 맵값이 정해진 블록들을 따로 모아서 각각의 마스터 부호책을 작성한다. 그러므로 유한상태 벡터 양자화기가 사용하는 마스터 부호책은 윤곽선을 많이 포함하는 블록들의 것과, 그렇지 않은 블록들의 것으로 두 가지로 작성된다.

각각의 마스터 부호책 작성 후에 상태 부호책의 설계는 영상의 이웃 블록들 사이의 상관관계를 이용하여 작성한다. 본 논문에서는 수직방향의 이웃 블록들 간의 상관관계를 유지하기 위하여 (그림 5)에서와 같이 부호단어 u를 열(column) 상태 변수로 사용하고, 수평방향의 이웃블록들 간 상관관계를 유지하기 위하여 l을 행(row) 상태 변수로 사용한다.



(그림 5) 유한상태 벡터 양자화기의 부호화. (블록크기 4 × 4)

(Fig. 5) Encoding of Finite State Vector Quantizer. (Block size 4 × 4)

열 상태 공간(space)은 $C=\{u|u$는 현재 입력 벡터 x에 대한 상위 블록의 부호단어$\}$이고, 행 상태 공간은 $R=\{l|l$은 현재 입력 벡터 x에 대한 좌측 블록의 부호단어$\}$이다. 각 블록의 크기가 $u \times v$일 때 마스터 부호책으로부터 상태 부호책 SC_s를 선택하는 알고리듬은 다음과 같다.

[단계 1] 마스터 부호책의 현재 부호단어 x에 대해 수평 방향의 왜곡(distortion),

$$hd(x, y)=\sum_{i=1}^{u}(x_{iv}-y_{i1})^2 \quad (1)$$

와 x에 대한 수직 방향 왜곡,

$$vd(x, y)=\sum_{j=1}^{v}(x_{uj}-y_{1j})^2 \quad (2)$$

을 계산하여, 두 값을 더한다. 즉

$$dist(x, y)=hd(x, y)+vd(x, y) \quad (3)$$

을 계산한다. 여기서 y는 x를 제외한 마스터 부호책의 부호단어이다.

[단계 2] x를 제외한 마스터 부호책의 모든 부호단어 y에 대해 [단계 1]의 $dist(x, y)$ 값을 계산한 후, 가장 작은 $dist(x, y)$ 값을 가지는 N_f개(상태 부호책의 크기)의 부호단어 y를 선택하여, 상태 부호책의 부호단어로 등록한다.

[단계 3] 마스터 부호책의 다음 부호단어를 x로 하여 더 이상의 부호단어가 없을 때까지 [단계 1]부터 반복한다.

본 논문에서 사용하는 상태 부호책 설계 후에 입력 영상의 부호화는 먼저 DCT 맵에 의해 현재 블록의 맵 값을 결정한 후, 맵 값에 따라 분리된 상태 부호책으로부터 현재 블록의 부호단어를 선택하여 복호화기로 전송한다. 상태 부호책으로부터 적합한 부호단어의 선택은 두 가지로 생각할 수 있다. 즉 먼저 현재 블록의 상위 블록과 좌측 블록의 상태를 이용하여 여러 상태 부호책 중에서 하나의 상태 부호책을 선택하고, 그 다음 선택된 현재 블록에 대해 가장 잘 부합되는 상태 부호책에서의 부호 단어 선택은 현재 블록의 픽셀값과 상태 부호책에 있는 각각의 부호단어들과의 왜곡을 계산하여 그 값이 가장 작은 값을 갖는 부호단어를 현재 블록의 부호단어로 결정하여 복호화기로 전송하는 것이다. 여기서 여러 상태 부호책 중에서 현재 블록에 대한 상태 부호책의 선택에는 마스터 부호책으로부터 상태 부호책을 설계할 때와 마찬가지의 방법을 사용한다.

현재 입력 벡터 x의 상태 부호책 SC_s 선택은 각 상

태 부호책의 대표 벡터 p와 현재 입력 벡터의 각각 이웃 열과 이웃 행을 따라서 x의 상위 블록 u와 좌측 블록 l에 가장 잘 부합되는 상태 부호책으로 한다. 즉, 크기 $m \times n$의 블록을 정의했을 때, 상태 부호책의 대표 벡터와 현재 입력 벡터의 상위 블록과의 수평 방향 왜곡 $hd(p, u)$를

$$hd(p, u) = \sum_{j=1}^{n} (p_{1j} - u_{mj})^2 \tag{4}$$

라하고, 수직 방향의 왜곡 $vd(p, l)$를

$$vd(p, l) = \sum_{i=1}^{m} (p_{i1} - l_{in})^2 \tag{5}$$

라 할 때, 입력 벡터 x의 왜곡 $dist(x)$는 다음과 같이 정의한다.

$$dist(x) = hd(p, u) + vd(p, l) \tag{6}$$

여러 상태 부호책 SC_s들의 대표 벡터 p에 대하여 가장 작은 왜곡 $dist$를 가지는 SC_s를 현재 입력 벡터의 상태 부호책으로 선택한다. 이와 같이 현재 입력 벡터에 대한 상태 부호책의 선택에 이전 블록의 정보를 이용하여 복호화기에서 상태 부호책의 선택을 위한 부가 정보를 부호화기에서 전송할 필요가 없게 된다. 이는 결국 큰 마스터 부호책에서 부호 단어를 선택하여 그에 대한 인덱스를 전송하는 것에 비하여 작은 크기의 상태 부호책의 인덱스만을 전송하여 비트율을 줄이는 효과를 가져온다.

6. 상태 부호책으로부터 부호단어의 선택 알고리듬

현재 입력 벡터 x의 상태 부호책 SC_s를 선택한 후에는 선택된 상태 부호책 안에 있는 여러 부호단어중에서 현재 입력 벡터에 가장 잘 부합되는 부호 단어를 선택해야 한다. 이는 입력 벡터 x와 상태 부호책의 각 부호단어의 유클리디언 왜곡을 계산하여 결정한다.

크기 $m \times n$의 블록을 정의했을 때, 입력 벡터 x는 $x = (x_{11}, x_{12}, ..., x_{1m}, ..., x_{mn})$로 나타낼 수 있다. 여기서 x_{mn}은 입력 벡터 x의 각 요소(픽셀)이다. 상태 부호책의 각 부호단어 y와의 왜곡은 식 (7)과 같이 정의한다.

$$dist(x, y) = \sum_{j=1}^{mn} (x_j - y_j)^2 \tag{7}$$

상태 부호책 SC_s의 각 부호단어 y에 대하여 가장 작은 왜곡 $dist$를 가지는 y의 인덱스를 선택하여 현재 입력 블록에 대한 부호 결과로서 복호화기로 그 인덱스를 전송한다.

7. 가변길이 부호구조

6절에서와 같이 상태 부호책에서 여러 부호 단어중 한 곳을 선택하여 그 부호 단어의 인덱스를 복호화기로 전송할 때, 본 논문에서는 이 인덱스의 부호화에 고정 길이 부호화 대신에 허프만 부호화를 사용한 가변 길이 부호화를 수행하는데 허프만 부호화를 할 때 부호생성을 쉽게 하고 속도를 높이기 위한 알고리듬을 제안한다.

7.1 부호 단어 길이(CWL) 표의 작성

부호책의 부호 단어 인덱스 집합 I를 정의하면, 집합 I는 $I = \{i_1, i_2, ..., i_n\}$이고, 각 인덱스 원소들에 대한 돗수분포 집합은 $P(i_j) = p_j (j = 1, 2, ..., n)$이고, 각 돗수분포가 $p_1 \geq p_2 \geq \cdots \geq p_n$라고 하면, 이와 같은 인덱스 집합 I가 주어졌을 때 I에 대한 부호 단어 길이(Code Word Length: CWL) 표의 작성을 위해 다음과 같은 알고리듬을 적용한다.

[단계 1] 기존 허프만 부호화[11] 방식과 같이 돗수분포에 따라 배열된 허프만 표에서 밑에서부터 두 인덱스를 묶어 나가면서 합산한 돗수값에 따라 새로운 병합 인덱스 c_j를 원래의 인덱스들 사이에 위치시킨다. 〈표 2〉에서 첫 열(column)의 i_5와 i_6 두 인덱스를 묶으면 새로운 병합 인덱스 c_1이 만들어지고 이 c_1 인덱스의 돗수값이 9이므로 두 번째 열의 i_4 밑에 놓이게 된다.

이와 같이 최종 두 인덱스만 남을 때까지 인덱스들을 줄여 나간다. 〈표 2〉의 작성 과정 중에 〈표 3〉의 CWL 표를 동시에 작성하는데 〈표 3〉에서 첫 열은 두 인덱스 합병시 돗수가 작은 것, 두 번째 열은 돗수가 큰 것이고, 세 번째 열은 그에 해당하는 부호길이를 나타내는 열로서 차후에 계산한다. 우선 〈표 2〉의 인덱스들을 합병해 나가면서 〈표 3〉의 CWL 표도 동시

에 작성한다. 이때 CWL 표에 대해 처리 속도를 높이기 위해 병합 인덱스 ($c_j; j=1, 2,...$)들에 대한 주소는 현재 j의 반전(inverse)된 값을 가지도록 한다. 예를 들어, 4 비트 워드로서 인덱스들을 나타낸다면 c_1은 0001의 반전값인 1110, c_2는 1101로 주소를 가지게 한다. 이렇게 하므로써 원(source) 인덱스 ($i_j; j=1, 2,...$)와 병합 인덱스와의 MSB(Most Significant Bit: 최상위 비트)만 검사하면 두 인덱스들을 쉽게 구별해 낼 수 있다.

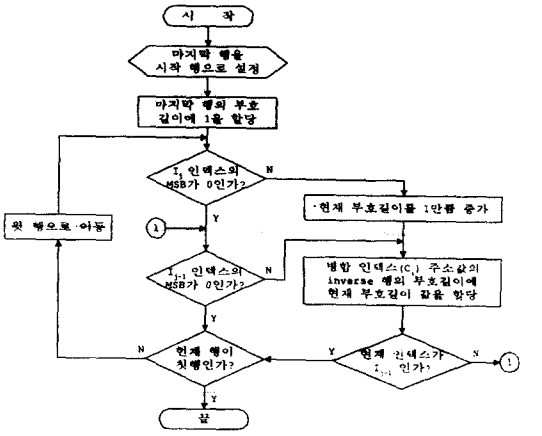
〈표 2〉 허프만 표의 예
〈Table 2〉 Example of huffman table

인덱스	도수	인덱스	도수	인덱스	도수	인덱스	도수	인덱스	도수
i_1	80	i_1	80	i_1	80	i_1	80	i_1	80
i_2	25	i_2	25	c_2	28	c_3	47	c_4	75
i_3	22	i_3	22	i_2	25	c_2	28		
i_4	19	i_4	19	i_3	22				
i_5	6	c_1	9						
i_6	3								

〈표 3〉 부호 단어 길이 표의 예
〈Table 3〉 Example of CWL table

행	i_j	i_{j-1}	부호길이
1	i_6	i_5	4
2	c_1	i_4	3
3	i_3	i_2	3
4	c_2	c_3	2
5	c_4	i_1	1

[단계 2] 〈표 3〉의 CWL 표에서 부호길이를 할당하는 알고리듬은 (그림 6)과 같다. CWL 표에서 원래의 인덱스와 병합 인덱스의 구분은 그 인덱스 주소의 MSB만을 검사하면 되므로 (그림 6)과 같은 알고리듬을 적용했을 때의 결과적인 부호길이가 〈표 3〉의 3열에 나타나 있다. 〈표 3〉에서 제 5행의 경우, 병합 인덱스 c_4의 MSB 비트가 1이므로 쉽게 검색이 가능하고 바로 이 인덱스의 주소 값을 반전(inverse)시킨 값인 4열에 부호 길이 2를 할당한다. 나머지 행의 부호 길이도 (그림 3)의 알고리듬에 따라 값을 할당한다.

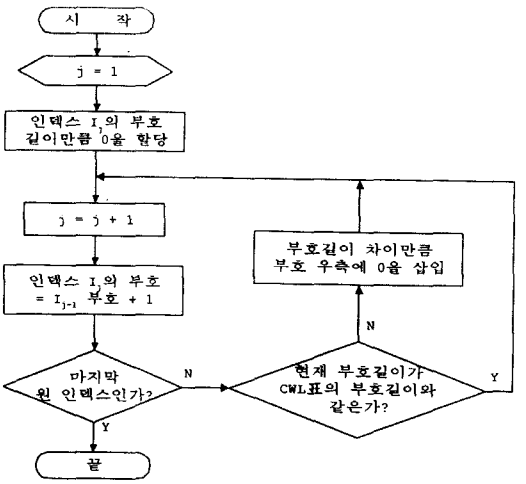


(그림 6) 부호 길이 할당 알고리듬의 흐름도
(Fig. 6) The flow chart of code word length assignment algorithm

7.2 단방향 분포 허프만 트리

기존의 허프만 부호화는 〈표 2〉와 같은 허프만 표에 따라 인덱스가 두개만 남을 때까지 병합시키고 다시 병합된 순서의 역으로 부호를 할당하기 때문에 부호 할당 알고리듬의 수행이 쉽지 않고 비교적 시간이 걸리는 편이다. 본 논문에서는 부호 생성을 빠르게 하기 위해 〈표 3〉에서의 부호길이가 결정되면 그 부호길이에 따라 허프만 부호화를 수행하는데, 단방향 분포(single-side distribution) 허프만 트리로 부호를 생성한다. (그림 7)에 부호생성 알고리듬의 흐름도가 나타나 있는데 그 흐름도에 따라서 〈표 3〉의 원(source) 인덱스(i_j, $j=1, 2,...$)에 대한 허프만 부호화를 수행한 결과가 〈표 4〉에 나타나 있다. 〈표 4〉에서 먼저 i_1 인덱스의 부호길이만큼 0을 할당하므로 허프만 부호는 "0"이 되고, 그 값에 1을 더한 값인 "1"을 i_2에 할당한 후, i_2의 부호 길이가 3이므로 "1" 뒤에 두개의 0을 더 추가하여 "100"으로 허프만 부호가 생성된다. 이와 같은 방법으로 나머지 인덱스에 대해서도 허프만 부호를 만든다.

이러한 단방향 분포 허프만 트리의 부호생성 알고리듬에서는 가장 짧은 인덱스의 부호는 "00.0"가 되고, 가장 긴 인덱스는 "11...1"의 부호로 만들어지므로 자기 오류 검출(self error checking) 기능을 가지고 있다. 즉 허프만 부호 생성과정의 이상여부를 알 수 있고 가장 긴 부호 길이의 마지막 원(source) 인덱스 부



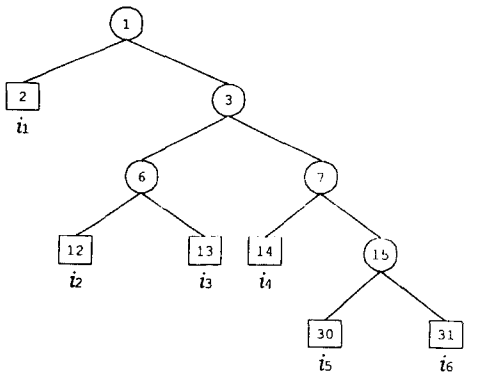
(그림 7) 단방향 분포 허프만 트리의 부호 생성 알고리듬 흐름도

(Fig. 7) The flow chart of code generation algorithm of single-side distribution huffman tree

〈표 4〉 허프만 부호 생성의 예

〈Table 4〉 Example of huffman code generation

부호길이	인덱스	허프만 부호
1	i_1	0
3	i_2	100
3	i_3	101
3	i_4	110
4	i_5	1110
4	i_6	1111



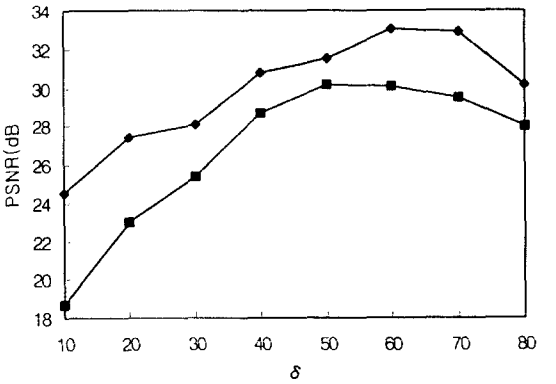
(그림 8) 표 4의 허프만 부호에 대한 단방향 분포 허프만 트리 구조

(Fig. 8) The single-side distribution huffman tree structure of huffman codes in Table 4

호를 검사하므로써 허프만 트리의 최종 부호가 생성되었는지를 쉽게 알 수 있다. (그림 8)은 〈표 4〉와 같이 허프만 부호가 생성된 인덱스들의 5단계의 단방향 분포 허프만 트리 구조를 보여주고 있다.

8. 실험결과

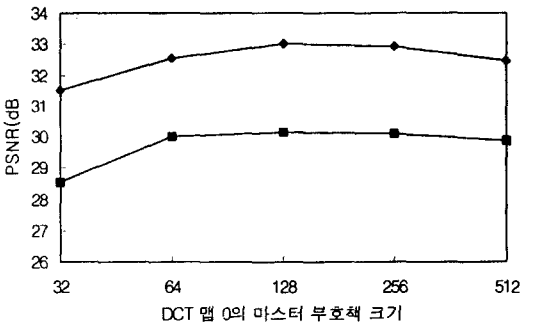
제안한 알고리듬의 평가는 512 × 512 크기의 Einstein과 Bridge 영상에 대해 수행하였고 모든 시뮬레이션은 SPARC 워크스테이션에서 수행하였다. 실험에 사용한 원 영상이 각각 (그림 13)의 (a)와 (b)에 나타나 있다. Einstein 영상은 비교적 윤곽선이 적고, Bridge 영상은 전체적으로 윤곽선 부분이 상당히 많은 영상으로 이는 윤곽선의 양에 의한 시뮬레이션 결과를 비교하기 위해서 선택하였다. (그림 9)는 윤곽선 영역(DCT 맵-"1") 마스터 부호책의 크기를 256으로, 비윤곽선 영역(DCT 맵-"0") 마스터 부호책의 크기를 128로 하고, 상태 부호책의 크기는 64로 했을 때 DCT 맵 임계값 δ를 10에서 80까지 변화시켰을 때 복원 영상의 화질을 비교한 것이다.



(그림 9) 마스터 부호책 크기가 128 (DCT 맵-"0"), 256 (DCT 맵-"1"), 상태 부호책의 크기를 64로 했을 때 DCT 맵 임계값 δ의 변화에 의한 복원 영상의 PSNR 비교 (◆ : Einstein 영상 ■ : Bridge 영상)

(Fig. 9) Comparison of PSNR for reconstructed image based on DCT map threshold δ at master codebook size 128 and 256, state codebook size 64(◆ : Einstein ■ : Bridge)

DCT 맵의 임계값인 δ는 블록의 평균 제곱 오차 값이 그 값보다 큰 블록에 대해서는 맵 값을 1로, 즉 윤곽선 영상으로 판별하는 역할을 한다. 그러므로 δ값이 커질수록 윤곽선 블록으로 선택하는 블록의 수는 그만큼 작아진다. (그림 9)에서 δ값이 커질수록 PSNR이 커져서 복원 영상의 화질이 좋아짐을 알 수 있다. Einstein 영상의 경우 δ값이 60일 때 가장 좋은 화질을, 70인 경우는 60인 경우보다는 약간 작지만 영상의 화질은 거의 같은 결과를 가져왔고, 80인 경우는 PSNR 값이 작아져서 δ를 계속 늘린다고 영상의 화질은 개선되지 않는 것을 볼 수 있다. 이는 δ의 큰 값으로 인해 DCT 맵을 설계할 때 거의 모든 블록들이 한쪽으로 맵핑되므로 그 후에 수행되는 두 개의 구별된 트리 구조 양자화기의 한쪽에 많은 벡터가 몰려 균형된 분류기로 동작되지 못하여 발생한 결과이다. Bridge 영상의 경우 Einstein 영상과 마찬가지로 δ 값이 커질수록 PSNR이 커져 50일 때가 가장 큰 값을 가지지만, δ가 50과 60 일때 PSNR 값이 거의 비슷한 것을 볼 수 있다. Bridge 영상이 Einstein 영상보다 윤곽선이 많이 존재하기 때문에 본 알고리듬의 수행결과 PSNR 값이 약간 적게 나타났고, 두 영상에 대해 모두 적당한 결과를 가지는 δ값을 60으로 하므로서 윤곽선 포함 여부에 비교적 덜 민감한 양자화기를 설계할 수 있다.

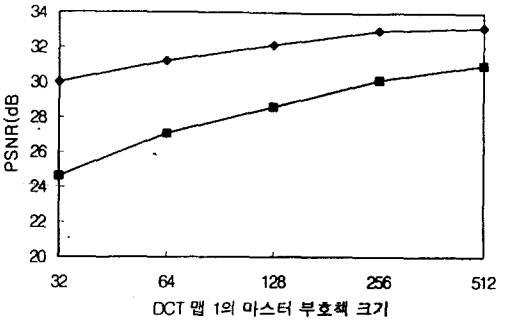


(그림 10) δ = 60, DCT 맵 1의 마스터 부호책 크기를 256으로, 상태 부호책 크기를 64로 했을 때 DCT 맵 0의 마스터 부호책 크기에 의한 복원 영상의 PSNR 비교. (◆ : Einstein 영상 ■ : Bridge 영상)

(Fig. 10) Comparison of PSNR for reconstructed image based on variable DCT map 0 master codebook size at δ = 60, DCT map 1 master codebook size 256, state codebook size 64 (◆ : Einstein ■ : Bridge)

(그림 10)은 δ를 60으로, DCT 맵 1의 마스터 부호책 크기를 256, 상태 부호책 크기를 64로 고정시키고 DCT 맵 0의 마스터 부호책 크기를 32부터 512로 했을 때 복원 영상의 PSNR을 비교한 것이다.

(그림 10)에서는 DCT 맵 0, 즉 블록 중에서 윤곽선이 적은 영역을 위한 마스터 부호책의 크기를 변화시켰을 때 복원 영상의 PSNR을 비교한 것으로, Einstein이나 Bridge 영상 모두에게서 PSNR의 큰 변화는 없었다. 이는 DCT 맵의 값이 0인 블록들은 비교적 영상에서 배경에 속하는 부분들이므로 그에 대한 배경 정보를 가지고 있는 마스터 부호책의 크기를 변화시켜도 전체적인 영상에는 별 영향이 없다는 것을 나타낸다. 그림에서 마스터 부호책의 크기가 128이나 256에서 두 영상 모두 비슷한 복원 영상의 화질을 나타냈는데, 마스터 부호책의 저장 용량 측면에서 크기를 128로 하는 것이 더 좋은 것을 알 수 있다.

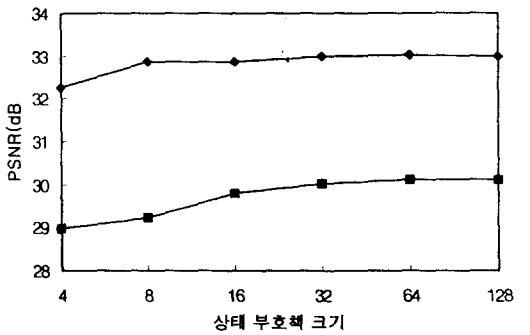


(그림 11) δ = 60, DCT 맵 0의 마스터 부호책 크기를 128로, 상태 부호책 크기를 64로 했을 때 DCT 맵 1의 마스터 부호책 크기에 의한 복원 영상의 PSNR 비교.(◆ : Einstein 영상 ■ : Bridge 영상)

(Fig. 11) Comparison of PSNR for reconstructed image based on variable DCT map 1 master codebook size at δ = 60, DCT map 0 master codebook size 128, state codebook size 64 (◆ : Einstein ■ : Bridge)

이번에는 윤곽선 정보를 포함하고 있는 DCT 맵 1인 블록을 위한 마스터 부호책의 크기를 변화시켰을 때 복원 영상의 화질을 조사한 것을 (그림 11)에 나타내었다. 이 경우는 영상의 윤곽선 정보가 복원 영상

의 화질을 큰 영향을 주기 때문에 PSNR값의 변화가 (그림 10) 보다는 컸다. 윤곽선 블록이 더 많은 Bridge 영상에서의 복원 영상 화질의 변화가 더 심했고, 마스터 부호책의 크기는 512일 때가 가장 영상의 화질이 좋았지만, 크기가 256일 때와는 큰 차이가 없었다. 이와 같은 결과로서 윤곽선에 대한 정보를 포함하고 있는 DCT 맵이 1인 마스터 부호책의 크기는 크게 해 주어야 각 영상의 윤곽선 정보를 잘 전송할 수 있는 것을 알 수 있다.



(그림 12) $\delta = 60$, 마스터 부호책 크기가 128 (DCT 맵-"0"), 256 (DCT 맵-"1") 일 때 상태 부호책 크기에 의한 복원 영상의 PSNR 비교.(◆ : Einstein 영상 ■ : Bridge 영상)

(Fig. 12) Comparison of PSNR for reconstructed image based on variable state codebook size at $\delta = 60$, master codebook size 128 and 256(◆ : Einstein ■ : Bridge)

(그림 12)는 δ를 60으로, 윤곽선 영역(DCT 맵-"1") 마스터 부호책의 크기를 256으로, 비윤곽선 영역 (DCT 맵-"0") 마스터 부호책의 크기를 128로 하고, 상태 부호책의 크기를 4부터 128까지 변화시켰을 때의 복원 영상 화질을 비교한 것이다. 상태 부호책의 설계는 5절에서 설명한 것과 같이 마스터 부호책으로부터 작성한 것이므로 결국 상태 부호책의 모든 부호단어는 마스터 부호책에 존재하는 것들이다. 그리고 마스터 부호책으로부터 상태 부호책을 설계할 때 마스터 부호책의 각 부호단어들에 대하여 식(3)과 같이 왜곡을 계산하여 그 값이 가장 작은 것부터 상태 부호책을 작성해 나가기 때문에 입력 벡터가 양자화기로 들어 왔을 때 각 상태 부호책에서 선택될 부호단어의 확률은 상태 부호책의 제일 위쪽에 있는 부호단어일수록 더 크다. (그림 12)에서 두 영상에 대해 상태 부호책의 크기를 변화시키면서 복원 영상의 화질을 비교해 봤을 때, PSNR 값이 거의 일정한 것을 볼 수 있다. 즉 대부분의 입력 벡터가 각 상태 부호책의 위쪽에 있는 부호단어로 양자화된다는 것을 알 수 있고, 그러므로 상태 부호책의 크기를 증가시켜도 PSNR 값의 변화는 크지 않고 거의 일정하다.

〈표 5〉 단방향 분포 허프만 트리 적용시의 비트율

〈Table 5〉 Bit rate of the single-side distribution huffman tree

영상	고정 길이 부호화	단방향 분포 허프만 트리를 이용한 가변 길이 부호화
Einstein	0.38 bpp	0.25 bpp
Bridge	0.38 bpp	0.28 bpp

〈표 5〉는 (그림 12)와 같은 조건에서 상태 부호책의 크기를 64로 했을 때 전송시의 비트율 비교이다. 상태 부호책에서 선택된 부호단어의 인덱스만을 복호화기로 전송하기 때문에 64개의 상태 부호책에 대해서는 입력 영상의 각 4×4 블록에 대해 6 비트로 전송한다. 그러나 7절에서 기술한 단방향 분포 허프만 트리를 이용한 가변 길이 부호화를 수행하면 전송 비트율을 줄일 수 있다. 〈표 5〉에서 각 실험 영상에 대한 그 결과가 나타나 있는데, 모두 고정 길이 부호화에 비하여 약 0.1 bpp 가량의 비트율을 감소시킬 수 있었다.

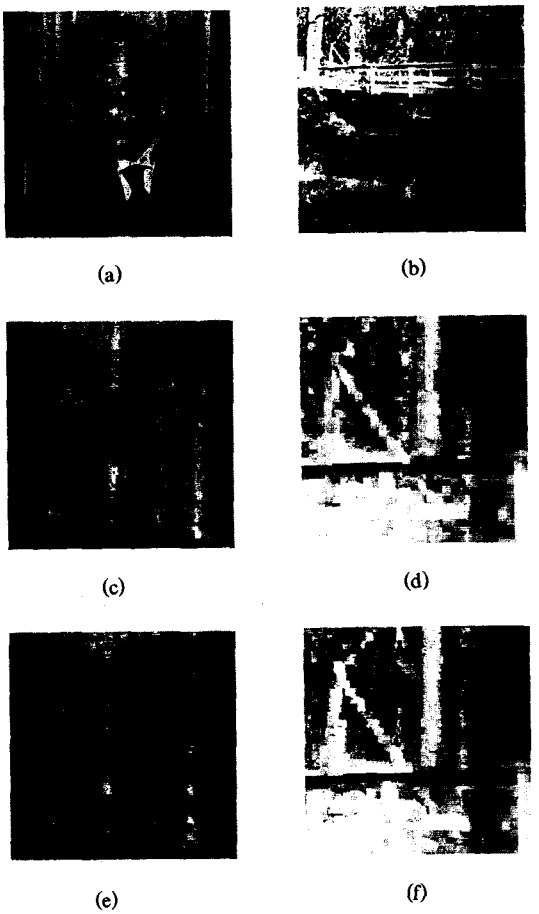
〈표 6〉 PNN, CVQ와 제안 알고리듬의 PSNR 비교

〈Table 6〉 Comparison of PSNR for PNN, CVQ and proposed algorithm

영상	Einstein	Bridge
	PSNR(dB)	PSNR(dB)
PNN	30.97	27.65
CVQ	31.26	29.14
제안 알고리듬	33.01	30.13

〈표 6〉에는 PNN[9]과 CVQ[10], 제안된 알고리듬의 PSNR값의 비교가 나타나 있다. Einstein 영상에 대해서는 PNN의 경우에 비해 PSNR은 2.04 dB, CVQ의 경우는 1.75 dB 증가하였고, Bridge 영상의 경우는 제안 알고리듬이 PSNR은 각각 2.48 dB, 0.99 dB 증가하여 본 논문에서 제안한 알고리듬이 PNN이나 CVQ에 비하여 더 좋은 결과를 나타내는 것을 알 수 있다.

(그림 13)에 Einstein과 Bridge 영상에 대한 PNN 알고리듬과 CVQ 알고리듬, 본 논문에서 제안한 알고리듬의 〈표 6〉에서의 시뮬레이션 결과가 나타나 있다. (a)와 (b)는 원 영상이고, 그 아래에는 각 알고리듬간의 비교를 확실히 하기 위해 영상의 한 부분을 확대한 그림이다. 그림에서 각각의 PNN, CVQ 결과 영상과 제안 알고리듬의 결과 영상을 비교해 보면 영상의 윤곽선(edge) 영역에서 다른 알고리듬보다 본 논문의 제안 알고리듬이 더 적게 왜곡이 일어난 것을 볼 수 있다.



(a) (b)

(c) (d)

(e) (f)

(g) (h)

(그림 13) "Einstein"과 "Bridge" 영상에 대한 PNN, CVQ, 제안 알고리듬의 비교
(a) "Einstein" 원 영상
(b) "Bridge" 원 영상
(c), (d) PNN의 결과 확대 영상
(e), (f) CVQ의 결과 확대 영상
(g), (h) 제안 알고리듬의 결과 확대 영상

(Fig. 13) Comparison of PNN, CVQ and proposed algorithm of "Einstein", "Bridge"
(a) "Einstein" Original image
(b) "Bridge" Original image
(c), (d) Reconstructed magnified image of PNN
(e), (f) Reconstructed magnified image of CVQ
(g), (h) Reconstructed magnified image of proposed algorithm

9. 결 론

본 논문에서는 영상의 벡터 양자화를 위한 새로운 부호책 설계 알고리듬을 제안하였다. 영상의 주파수 성분에 따라 DCT 맵을 작성하여, 각 블록의 2차원 DCT 계수 값을 이용하여 특징을 추출하고 블록의 맵 값에 따라 따로 부호책을 설계하였다. 영상의 블록간 상관관계를 이용하는 유한상태 벡터 양자화기를 이용하여 부호화를 수행하기 위해 마스터 부호책의 설계에 2차원 DCT의 계수 값의 특징을 원 영상으로부터 추출하고, 각 마스터 부호책의 부호단어(codeword)를 작성하기 위해 이진(binary) 트리 분류기를 사용하였다. 각 맵에 따른 상태 부호책은 마스터 부호책으로부터 각 부호단어의 왜곡을 계산하여 설계하였으며, 복호화기로 부호책의 각 부호단어에 대한 인덱스

를 전송할 때에 그 인덱스는 허프만 부호의 검색 속도를 빠르게 하기 위하여 단방향 분포 허프만 트리를 이용하여 부호화를 수행하였다. 시뮬레이션 결과 본 논문에서 제안한 방법으로 영상 부호화를 수행했을 경우 PNN이나 CVQ 알고리듬과 비교하여 더 좋은 화질의 영상을 얻을 수 있었다.

참 고 문 헌

[1] J.Makhoul et al., "Vector quantization in speech coding," Proc. IEEE, vol.73, pp.1551-1588, Nov. 1985.

[2] N.M.Nasrabadi and R.B.King, "Image coding using vector quantization : a review," IEEE Trans. Commun., vol. COM-36, pp. 957-971, Aug. 1988.

[3] R.M.Gray, "Vector quantization," IEEE ASSP Mag., vol. 1, pp. 4-29, April 1984.

[4] A.Gersho and R.M.Gray, *Vector Quantization and Signal Compression*. Boston : Kluwer Academic Publishers, 1992.

[5] S.P.Lloyd, "Least-squares quantization in PCM," IEEE Trans. Inform. Theory, vol. IT-28, pp. 129-137, March 1982.

[6] J.A.Hartigan, *Clustering Algorithm*. New York : Wiley, 1975.

[7] Y.Linde, A.Buzo, and R.M.Gray, "An algorithm for vector quantizer design," IEEE Trans. Commun., vol. COM-28, pp. 84-95, Jan. 1980.

[8] W.Equitz, "Fast algorithm for vector quantization picture coding," IEEE Int. Conf. on ASSP, pp. 725-727, 1987.

[9] W.Equitz, "A new vector quantization clustering algorithm," IEEE Trans. ASSP, vol. 37, pp. 1568-1575, Oct. 1989.

[10] B.Ramamurthi and A.Gersho, "Classified vector quantization of images," IEEE Trans. Commun., vol. COM-34, no. 11, pp. 1105-1115, Nov. 1986.

[11] D.A.Huffman, "A Method for the construction of minimun redundancy codes," Proc. IRE, vol. 40, no. 10, pp. 1098-1101, Sept. 1952.

[12] S.Roman, *Coding and Information Theory*, Springer-Verlag, New York, 1992.

[13] A. Buzo et al., "Speech coding based upon vector quantization," IEEE Trans. Acoust., Speech, Signal Processing, vol. ASSP-28, pp. 562-574, Oct. 1980.

[14] E.A.Riskin and R.M.Gray, "Lookahead in growing tree-structured vector quantizers," IEEE Int. Conf. on ASSP, pp. 2989-2292, Mar. 1991.

[15] P.A.Chou, T.Lookabaugh, and R.M.Gray, "Optimal pruning with applications to tree-structured source coding and modeling," IEEE Trans. Inform. Theory, vol. 35, pp. 299-316, Mar. 1989.

[16] H.S.Hou, "A fast recursive algorithm for computing the discrete cosine transform," IEEE Trans. Acoust., Speech, Signal Processing, vol. ASSP-35, pp. 1455-1461, Oct. 1987.

[17] W.H.Chen, C.H.Smith, and S.C.Fralick, "A fast computational algorithm for the discrete cosine transform," IEEE Trans. Commun., vol. COM-25, pp. 1004-1008, Sept. 1977.

[18] B.G.Lee, "A new algorithm to compute the discrete cosine transform," IEEE Trans. Acoust., Speech, Signal Processing, vol. ASSP-32, pp. 1243-1245, Dec. 1984.

[19] W.H.Chen and W.K.Pratt, "Scene adaptive coder," IEEE Trans. Commun., vol. COM-32, pp. 225-232, Mar. 1984.

[20] J.Foster, R.M.Gray and M.O.Dunham, "Finite-state vector quantization for waveform coding," IEEE Transaction Information Theory, vol. IT-31, pp. 348-359, May 1985.

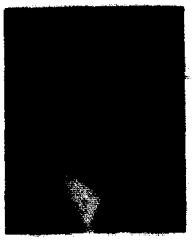
[21] R.L.Bker and H.H.Shen, "A finite-state vector quantizer for low-rate image sequence coding," in Proc. ICASSP, 1987, pp. 760-763.

[22] C.S.Kim, J.Bruder, M.J.T.Smith, and R.M. Mersereau, "Subband coding of color images using finite state vector quantization," in Proc. ICASSP, 1988, pp. 753-756.

[23] N.M.Nasraba and Y.Feng, "A dynamic finite-state vector quantization scheme," in Proc. ICASSP,

1990, pp. 2261-2264.

[24] T.Kim, "New finite state vector quantizers for images," in Proc. ICASSP, 1988, pp. 1180-1183.



조 성 환

1980년 성균관대학교 전자공학과 졸업(학사)
1982년 성균관대학교 대학원 전자공학과(공학석사)
1991년 성균관대학교 대학원 전자공학과(공학박사)
1982년~1985년 해군사관학교 전기공학과 전임강사
1997년 미국 Columbia 대학 Visiting Scholar
1985년~현재 대유공업전문대학 전자계산기과 부교수
관심분야: 영상처리, 신경회로망, 패턴인식