Journal of Korea Multimedia Society Vol. 15, No. 4, April 2012(pp. 485-491)

HEVC의 분할 영역에서 효율적인 움직임 정보 표현

이동식[†], 김영모[††]

요 약

본 논문은 움직임 벡터와 함께 Coding Unit (CU)의 분할 정보를 표현하기 위해 쿼드트리 기반의 Coding Unit Tree (CUT)를 제안한다. 새로운 동영상 국제 표준안인 High Efficiency Video Coding (HEVC)는 높은 압축 효율을 위해 다양한 새로운 기술들을 채택하였다. 그리고 CU, prediction Unit (PU), 와 Transform Unit (TU)라는 분할 개념을 도입하였다. 그중 기본 부호화 단위인 CU는 H.264/AVC의 매크로 블록보다 다양한 크기를 제공하며 계층적인 구조를 가지고 있으며 쿼드트리 기반의 영상을 분할하고 처리한다. 이러한 구조는 유연성과 최적화를 이룰 수 있는 기반을 제공하고 있으나, 분할 정보에 대한 오버헤더가 발생한다. 복잡한 움직임 정보가 발생하면, 해당하는 정보를 전송하기 위해 다양한 신호가 발생한다. 본 논문에서는 이러한 다양한 신호들을 분석하고, 중복되는 정보를 제거하기 위한 알고리즘을 제안한다. 제안하는 알고리즘은 기본 블록인 2x2 블록을 기준으로 계층적인 구조를 제안한다. 제안하는 알고리즘은 쿼드트리 기반의 타입 코드로 영상을 구조를 나타내고, 대표 값과 잔여 값으로 각 노드의 값을 표현한다. 결과에서 제안하는 알고리즘이 HM1.0보다 13.6% 압축 향상을 보여준다.

Efficient Motion Information Representation in Splitting Region of HEVC

Dong-Shik Lee[†], Young-Mo Kim[††]

ABSTRACT

This paper proposes 'Coding Unit Tree' based on quadtree efficiently with motion vector to represent splitting information of a Coding Unit (CU) in HEVC. The new international video coding, High Efficiency Video Coding (HEVC), adopts various techniques and new unit concept: CU, Prediction Unit (PU), and Transform Unit (TU). The basic coding unit, CU is larger than macroblock of H.264/AVC and it splits to process image-based quadtree with a hierarchical structure. However, in case that there are complex motions in CU, the more signaling bits with motion information need to be transmitted. This structure provides a flexibility and a base for a optimization, but there are overhead about splitting information. This paper analyzes those signals and proposes a new algorithm which removes those redundancy. The proposed algorithm utilizes a type code, a dominant value, and residue values at a node in quadtree to remove the addition bits. Type code represents a structure of an image tree and the two values represent a node value. The results show that the proposed algorithm gains 13.6% bit-rate reduction over the HM-1.0.

Key words: HEVC, Coding Unit(부호화 단위), Split block(분할 블록), Quadtree(쿼드트리), Motion information(움직임 정보)

※ 교신저자(Corresponding Author) : 김영모, 주소 : 대구광역시 북구 산격동 1370 경북대학교 공대 10호관 615호 (702-701), 전화 : 011)528-9864, FAX : 0505)959-5541, E-mail : ymkim@ee.knu.ac.kr
접수일 : 2011년 11월 15일, 수정일 : 2011년 12월 16일
완료일 : 2012년 1월 25일
[†] 정회원, 경북대학교 전자전기컴퓨터학부 (E-mail : lds@ee.knu.ac.kr)
[††] 정회원, 경북대학교 전자전기컴퓨터학부

1. 서 론

ISO/IEC MPEG과 ITU-T VCEG의 공동 작업반인 Joint Collaboration Team on Video Coding (JCT-VC)에서는 초고해상도(UHD: Ultra HD) 동영상 부호화를 주 목표로 한 HEVC(High Efficiency Video Coding) 표준화를 진행하고 있다. 지난 2010년 10월 광저우에서 개최된 3차 회의에 서는 HEVC의 참조 소프트웨어인 HM(HEVC test Model) 1.0에 포함될 기술이 결정되었고, 2010년 3월 제네바 5차 회의에 걸쳐 HM 3.0에 포함될 기술이 채택되었다. HEVC의 표준화 단계에서는 공통 실험조건을 명시하고 HM의 부호화 모드로 Intra-only, 저지연(Low-Delay: LD), 그리고 임의접근(Random Access: RA)의 3가지 모드를 정의하고 있다.

JCT-VC는 2010년 10월에 HEVC라 불리는 차세대 부호화 표준의 테스트 모델로 HM (Tmuc)를 표준화하였다[1, 10]. 이 테스트 모델에서 나무 구조 부호화 단위 (Coding Unit : CU)가 채택되었고, 이 나무 구조 CU는 그림 1에서 보여주듯 8×8(N_3)~64×64(N_0) 크기의 밝기 값이다. 그림 1은 나무 구조 CU와 PU를 보여주고 있다.

이 구조는 H.264의 가변 블록 크기 움직임 보상과 달리, 깊이 4를 가지는 쿼드트리 기반의 계층적인 구조를 가진다. 그림 1의 (b)는 예측 단위 (Prediction Unit : PU)의 부호화 과정을 보여준다. TU 는 깊이

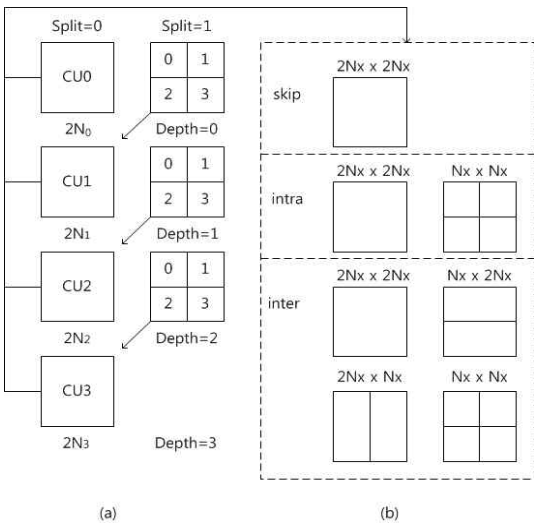

그림 1. CU와 PU 구조

4를 가지고 4×4부터 32×32까지 변한다. 이들 단위들은 나무 구조 단위라고 불리며 다양한 움직임 추정/보상이 가능하게 해준다. 이것은 움직임 보상 이후 잔여 데이터에서의 중복을 제거하는 것을 목적으로 개발되었다. 이 구조는 이전 표준안에 비해 큰 단위 블록(64×64)부터 작은 단위 블록(4×4)까지 사용하여 영상에서 발생하는 다양한 크기의 움직임들을 보상하게 해 준다. 이 구조는 영상에서 자세하고 다양한 움직임을 탐지해 주어 H.264와 MPEG-4에 비해 같은 화질에 대해 더 좋은 압축 성능을 제공해 준다.

그러나, 더 좋은 움직임 탐지가 제공되지만 높은 복잡도는 느린 부호화 속도를 유발시킨다. 또한 향상된 기술들은 잔여 데이터를 줄여주지만, 블록 정보는 오히려 증가시킨다. HEVC에서 블록의 움직임 분석을 통한 새로운 블록 단위들의 개념, 분할 크기, 통합 블록, 그리고 속도 향상을 위한 다양한 연구들이 있다. Leng 등[2]은 CU의 발생 빈도를 분석하여 해당 깊이에서 CU를 건너뛰어 부호화기의 속도를 향상시킨다. Ugur 등[3]은 64×64, 32×32, 16×16, 8×16, 16×8, 및 8×8 분할 크기의 매크로 블록으로 쿼드트리 기반의 부호화 구조를 사용하였다. Oudin과 Marpe 등[4, 5]은 영상이 예측 블록으로 과도하게 분열되는 것을 방지하고 예측 신호를 위한 비트 수를 줄이기 위해 단말 통합을 통한 비트율-왜곡 최적화를 제안하였다. Marpe [5]는 특별히 Coding Tree Block (CTB)라는 매크로 블록의 확장 개념과 영상 분할이라는 개념을 도입하였다. Han 등[6]은 세 가지 블록 개념인 CU, PU와 TU라는 높은 유연성을 가지는 계층성을 제안하였다. 세 가지 다른 개념들로 분할되는 이 블록 구조는 각각을 최적화 시킬 수 있다. Marpe [7]는 내장(nested)되고 미리 정의된 쿼드트리 구조를 사용하여 예측과 변환을 위한 블록 분할의 유연성을 도입하였다.

이러한 구조는 유연성과 최적화를 이룰 수 있는 기반을 제공하고 있으나, 분할 정보에 대한 오버헤더가 발생한다. 복잡한 움직임 정보가 발생하면, 해당하는 정보를 전송하기 위해 다양한 신호가 발생한다. 본 논문은 이러한 문제점을 해결하면서 CU를 효율적으로 표현하는 새로운 구조를 제안하고 이것을 Coding Unit Tree (CUT) 라고 부른다. 이 구조는 회색 상보 쿼드트리 구조 [9, 11]를 채택한다. 최하위 레벨에서 4개의 자식 노드에서 타입 코드, 대표 값과

잔여 값들을 결정한다. 결정된 타입 코드와 잔여 값들은 각각의 파일에 저장되고, 대표 값은 상위 레벨의 상위 노드에 전송된다. 이러한 절차는 최상위 레벨까지 수행되고 마지막 대표 값은 저장된다. 부호화 절차 후에, 타입 코드는 엔트로피 부호화기에 의해 부호화된다.

2. HEVC의 나무 구조 CU

HEVC는 표 1과 같은 부호화 파라미터들을 가진다. HEVC는 계층적인 쿼드트리를 가지고 4×4~64×64까지의 움직임 벡터를 만들 수 있다. 그러므로 2N×2N의 64×64의 CU에서 16×16 크기의 PU들, 즉 256개의 PU들과 256개의 움직임 벡터들이 발생할 수 있다. Depth, Merging bit, Split bit, Part size, and Motion vector difference (mvd)들이 움직임 정보의 구조를 나타낸다. 그림 2에서는 네 개의 동영상 각각에서 8개의 P-slice에서 발생하는 잔여 데이터의 계수와 움직임 정보를 포함하는 헤더 정보의 비율을 보여준다. 그림에서 헤더 정보의 비율이 잔여 데이터의 계수의 비율에 비해 무시하지 못할 정도인 것을 알 수 있다.

표 1 부호화 파라미터들

Frame-level parameters	frame type, index, QP, resolution, coding unit size
CTB partition	CTB partition, filter block partition
CTB-level parameter	intra prediction mode, motion vector, reference index QP

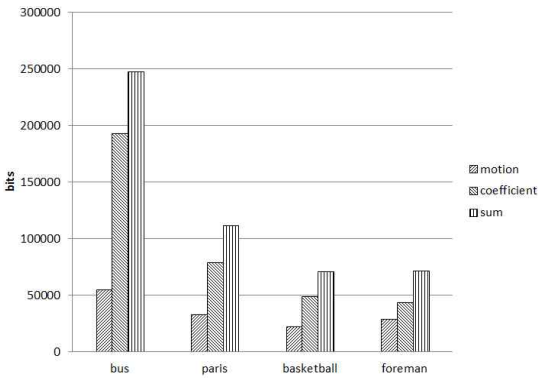

그림 2 잔여 데이터의 계수와 움직임 정보의 상대적인 비율

3. Covering Quadtree

우리는 특별한 쿼드트리 분할에 의한 가능한 블록들을 고려하면서 좀 더 현실적인 문제에 집중한다. 그림 3에서와 같이 포함 쿼드트리는 각 노드가 4개 또는 0개의 자식을 가지는 쿼드트리다. 포함 쿼드트리는 다음과 같은 특징들을 갖는다. 1) 영상 나무에서 각 노드는 정사각형 블록에 해당한다. 2) 블록의 네 자식 블록은 네 개의 작은 크기의 중첩되지 않는 정사각형 하위 블록이며 블록은 포함한다. 3) 루트 노드는 전체 영상을 포함한다. 결과 블록들은 포함 쿼드트리의 단말 노드들에 해당한다. 쿼드트리는 이 차원 영상 처리에서 유용한데, 그 이유는 블록 구성 부호화의 간단함과 효율성 때문이며 나무의 각 노드 당 1비트만을 요구한다.

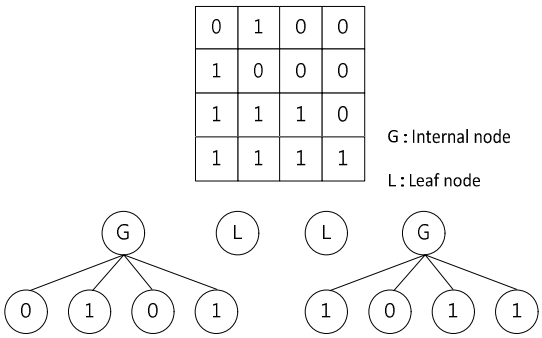

그림 3. Covering Quadtree.

4. 제안하는 구조 : CUT

간단하게 설명하기 위해, 고려중인 영상들은 N×N 화소이며, N은 2의 배수이다. 각 화소는 b 비트를 포함한다. 왼쪽 위 화소에 (0, 0)의 (X, Y) 축을 할당하고, X축은 오른쪽을 증가하고 Y축은 아래쪽으로 증가한다. CUT는 쿼드트리의 노드를 나타내는 대표 값, 잔여 값과 타입 코드를 가지고 영상을 잘 압축한다. 4 자식 노드는 움직임 방향에 따라 이름이 붙여진다: NE, SW, SE와 SW. 제안하는 알고리즘의 부호화 처리는 아래와 같다.

1) 그림 4에 따라 각 노드에서 대표 값은 부모 노드에 할당된다. 타입 코드와 잔여 값을 각각 파일에 저장된다.

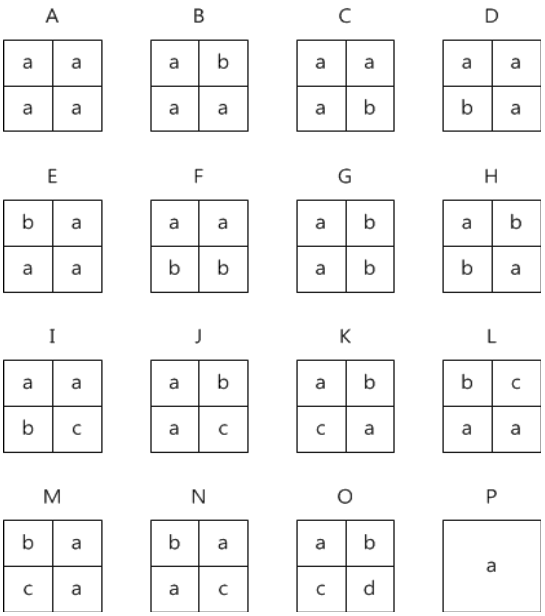

그림 4. 제안하는 타입 코드들 : 대문자는 타입 코드를 나타낸다. 소문자는 mvd를 나타낸다.

2) 최상위 노드까지 1을 반복한다.

복호화 과정은 위의 부호화 과정의 반대이다. 우리는 영상 나무를 나타내기 위해 세 가지 심볼들이 필요하다. 그리고 이들은 아래에서 설명한다.

4.1 대표 값과 잔여 값

대표 값과 잔여 값은 영상 나무에서 모든 자식 노드들의 값을 나타내기 위해 필요하다.

단말 노드는 자식이 없으며, 내부 노드(즉, 비 단말 노드)는 네 개의 포인터를 각각 가진다. 입력 영상의 화소 값은 단말 노드에 할당이 되고, 부모 노드들의 값은 네 개의 자식 노드들에서 영향을 받는다. 네 개의 자식 노드들의 가장 빈번한 값이 노드의 값으로 결정되어 진다. 즉 대표 값이다. 만약, 네 개의 자식 노드의 값 중 가장 빈번한 값이 없다면, 그림 4에 따라 대표 값을 결정한다. 노드의 잔여 값은 대표 값과 서도 다른 값들과의 차이 값이다. 우리는 bottom-up 방식을 사용하여 단말에서 최상위 노드까지 영상의 나무를 만든다.

4.2 타입 코드

타입 코드는 자체가 기본 블록, 2x2 블록의 모양을 나타낸다 (그림 4). 소문자 (‘a’ 부터 ‘d’) 는 mvd를 나타내며, 대문자는 타입 코드를 각각 나타낸다. 타입 코드들의 결과는 계층적으로 영상을 나타나내는 블록들의 모양을 만들어 낸다. 이 타입 코드는 16개의 타입 코드로 이루어졌으며, 단말 노드 타입 (‘P’)를 포함하고 있으며, 그림 4에 나타내져 있다. Li [8]은 81개의 타입 코드를 제안했지만, 본 논문은 15개와 한 개의 단말 타입 코드를 제안하고 있다.

예를들어, ‘A’는 네 개의 자식 노드들이 같은 값을 가지는 것을 의미하며, 나머지 타입 코드들은 각각의 모양을 의미한다. 그리고 ‘P’는 자체로 단말 노드를 가리키거나, 네 개의 자식 노드들이 같은 값을 가지는 단말 노드들이다.

4.3 CUT 예제

그레이 상보 쿼드트리는 각 자식 노드들의 값의 분포를 분석한 후 대표 값과 잔여 값으로 노드의 값을 표현하고, 영상 나무의 구조는 타입 코드로 표현한다. 그림 5의 예제는 Foreman에서 가져온 mvd 값들이다.

그림 5를 CUT로 표현하면 ‘A0, J31, P, P, F1’와 같다. 최하위 레벨의 기본 블록에서 부터 시작하여 상위 레벨의 노드로 대표 값을 전달하고, 타입 코드와 잔여 값들은 각각의 파일에 저장한다. 그림 6은 제안하는 알고리즘의 플로우 차트를 보여준다. 매크로 블록에서 비트율-왜곡 최적화 (rate-distortion optimization) 계산 결과에 따라 파티션을 나누고 첫 번째 파티션 정보를 읽어 들인다. 파티션의 타입 코드를 헤더에 저장하고 타입 코드에 따라 필요한 대표 mvd값과 나머지 값을 계산한 다음, 계산된 값들을 저장한다. 다음 파티션 정보를 읽어 들이고 위의 수행을 반복한다.

0	-3	0	0
0	-1	0	0
0	0	0	0
0	0	-1	-1

그림 5. Foreman 영상 열에서 발생한 움직임 벡터들

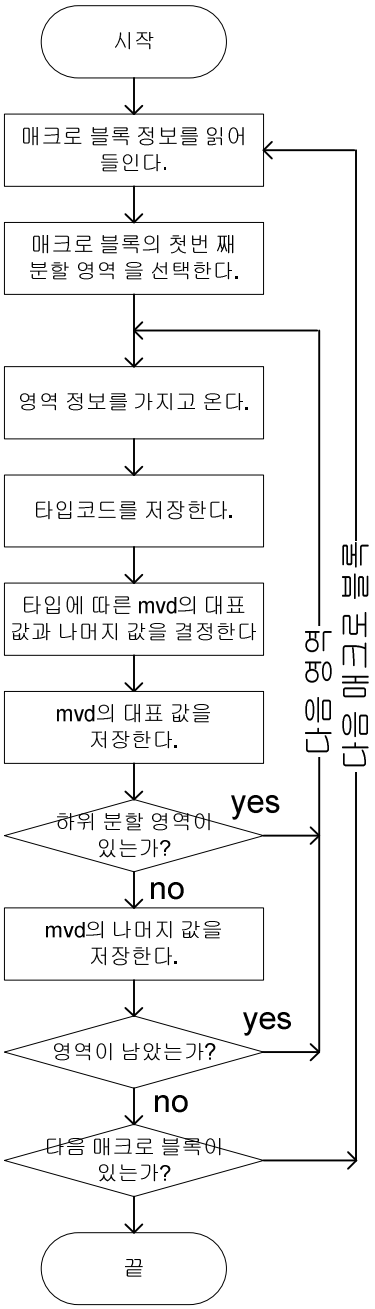

그림 6. 플로우 차트

4.4 HEVC에서의 예제

예를들어, mvd가 '00 00 00, 00 00 00 01' 일때, 이들 코드들은 HEVC의 헤더에서 'Depth, Merging, Split bit, Part size, N×N, 00 01'로 나타내어진다. 그러나 CUT에서는 'P0 A0 PPPC1'로 나타낸다. 'P0'에서, 'P'는 타입 코드를 나타내고, '0'은 x축에서의 mvd를 나타낸다. 그리고 'A0 PPPC1'은 y축 성분이다. 'A'와 'C'는 각각 타입 코드들이고, 0은 최상위 노드의 대표 값이고, 1은 mvd의 잔여 값이다. 'P0 A0'는 2N×2N 깊이에서, 'PPPC1'은 N×N 깊이에서 타입코드, 대표 값과 잔여 값을 나타낸다. 그림 3에서, 'C'는 mvd의 대표 값과 잔여 값 1개가 필요하며, 최상위 노드가 아닌 경우는 대표 값은 상위 노드로 전송된다.

주어진 타입 코드는 N-1 레벨에서는 'A' 타입 코드가 발생하지 않기 때문에, 15개의 심볼들이 적용된 산술 부호화기를 사용하며, 나머지 레벨에서는 16개의 심볼들이 적용된 산술 부호화기를 사용한다. 대표 값과 잔여 값은 일반적인 산술 부호화기를 사용하여 부호화 한다.

5. 실험 결과

우리는 part size, split information, and mvd 등을 포함하는 헤더 정보를 얻기 위해 JCT-VC에서 제공하는 HM1.0 소프트웨어를 사용하였다. 실험에 사용된 영상은 QCIF (Foreman, Bus, and Paris)와 416×240(Basketball)와 같은 두 종류의 해상도를 가진다. 사용된 프로파일은 MaxCUWidth와 Height는 64이고, Depth는 4, inter-frame으로는 P-slice만을 사용하였다. Context adaptive binary arithmetic coding (CABAC)과 Hadamard변환을 사용하였다. 제안하는 알고리즘의 성능은 수식 (1)로 테스트된다.

$$\Delta Bitrate = (Bitrate_{HM} - Bitrate_{CUT}) / Bitrate_{HM} \quad (1)$$

HM software와 CUT의 비트 율 비교는 표 2에서 5에서 표시하였고, 이 실험들에서 CUT의 엔트로피 부호화기는 심볼 수에 따른 산술 부호화기를 사용하였다. 표 2에서 5까지는 양자화 파라미터 (QP)에 따른 압축률의 결과를 보여주고 있다. 압축률이 낮은 QP 24에서는 움직임 정보에 대한 비트율이 높게 나타남과 동시에 CUT의 압축률도 좋게 나타난다. 제안하는 알고리즘은 HM1.0 보다 평균 13.6% 비트율 감소를 제공한다.

우리는 헤더 정보에서 중복된 정보를 제거하는 새로운 알고리즘을 제안한다. 실험 결과에서 제안하는 알고리즘이 헤더의 비트 율을 감소시키는 것을 볼 수 있다.

표 2. HM software와 CUT의 움직임 정보 압축 결과 비교 : QP 24

	HM1.0 (bits)	CUT (bits)	Reduction (%)
Bus	64,932	53,176	18.1
Paris	38,724	35,200	9.1
Basketball	27,825	24,288	12.7
Foreman	41,254	33,936	17.7

표 3. HM software와 CUT의 움직임 정보 압축 결과 비교 : QP 28

	HM1.0 (bits)	CUT (bits)	Reduction (%)
Bus	54,395	44,520	18.2
Paris	32,212	29,536	8.3
Basketball	21,690	18,672	13.9
Foreman	28,003	23,512	16.0

표 4. HM software와 CUT의 움직임 정보 압축 결과 비교 : QP 32

	HM1.0 (bits)	CUT (bits)	Reduction (%)
Bus	41,825	34,816	16.8
Paris	25,398	24,032	5.4
Basketball	15,900	14,504	8.8
Foreman	16,456	13,744	16.5

표 5. HM software와 CUT의 움직임 정보 압축 결과 비교 : QP 36

	HM1.0 (bits)	CUT (bits)	Reduction (%)
Bus	29,017	24,192	16.6
Paris	17,145	16,160	5.7
Basketball	11,015	10,568	4.1
Foreman	9,628	7,808	18.9

5. 결 론

새로운 압축 표준안인 HEVC는 보다 좋은 압축 성능을 제공하고 있지만 채택하고 있는 기술들로 인해, 오히려 헤더가 전체 비트열에서 큰 비율을 차지한다. 복잡한 움직임이 발생할 경우, 움직임 정보를 나타내기 위한 여러 가지 비트들이 반복적으로 발생한다. 이러한 비트들의 중복성을 제거하기 위해, 제안하는 알고리즘은 16개의 타입 코드를 정의하여 기본 블록인 2x2 블록의 기본 모양을 나타낸다. 그리고 대표 값과 잔여 값들로 쿼드트리 구조에서 노드의 값을 나타낸다. 제안하는 알고리즘은 개념적으로 간단하고 유연하며, HM1.0 보다 더 좋은 성능을 보여준다.

참 고 문 헌

[1] Joint Collaborative Team: *Video Coding, Video Coding Technology Proposal by Samsung (and bbc)*, JCTVC-A124, Dresden, Germany, 2101.

[2] Jie Leng, Lei Sun, Takeshi Ikenaga, and Shinichi Sakaida, “Content Based Hierarchical Fast Coding Unit Decision Algorithm for HEVC,” *2011 International Conference onMultimedia and Signal Processing,* Vol.1, pp. 56-59, 2011.

[3] Kemal Ugur, Kenneth Andersson, Arild Fuldseth, Gisle Bjøntegaard, Lars Petter Endresen, Jani Lainema, Antti Hallapuro, Justin Ridge, Dmytro Rusanovskyy, Cixun Zhang, Andrey Norkin, Clinton Priddle, Thomas Rusert, Jonatan Samuelsson, Rickard Sj¨oberg, and Zhuangfei Wu, “High Performance, Low Complexity Video Coding and the Emerging Hevc Standard,” *IEEE Transactions on Circuits and Systems for Video Technology,* Vol.20, No.12, pp. 1688-1697, 2010.

[4] S. Oudin, P. Helle, J. Stegemann, C. Bartnik, B. Bross, D. Marpe, H. Schwarz, and T. Wiegand, “Block Merging for Quadtree-Based Video Coding,” *2011 IEEE International Conference on Multimedia and Expo,* pp. 1-6, 2011.

[5] Detlev Marpe, Heiko Schwarz, Sebastian Bosse, Benjamin Bross, Philipp Helle, Tobias Hinz, Heiner Kirchhoffer, Haricharan Lakshman, Tung Nguyen, Simon Oudin, Mischa Siekmann, Karsten S¨uhring, Martin Winken, and Thomas Wiegand, “Video Compression Using Nested Quadtree Structures, Leaf Merg-

ing, and Improved Techniques for Motion Representation and Entropy Coding," *IEEE Transactions on Circuits and Systems for Video Technology,* Vol.20, pp. No.12, 1676-1687, 2010.

[6] Woo-Jin Han, Junghye Min, Il-Koo Kim, Elena Alshina, Alexander Alshin, Tammy Lee, Jianle Chen, Vadim Seregin, Sunil Lee, Yoon Mi Hong, Min-Su Cheon, Nikolay Shlyakhov, Ken McCann, Thomas Davies, and Jeong-Hoon Park, "Improved Video Compression Efficiency Through Flexible Unit Representation and Corresponding Extension of Coding Tools," *IEEE Transactions on Circuits and Systems for Video Technology,* Vol.20, No.12, pp. 1709-1720, 2010.

[7] Detlev Marpe, Heiko Schwarz, Sebastian Bosse, Benjamin Bross, Philipp Helle, Tobias Hinz, Heiner Kirchhoffer, Haricharan Lakshman, Tung Nguyen, Simon Oudin, Mischa Siekmann, Karsten Sühring, Martin Winken, and Thomas Wiegand, "Highly Efficient Video Compression Using Quadtree Structures and Improved Techniques for Motion Representation and Entropy Coding," *2010 Picture Coding Symposium*, pp. 206-209, 2010.

[8] Zhongqiang Li and Duncan Telfer, "Primitive Quadtree and Type Code Quadtree Approaches for the Representation of Binary Regions," *IEE Colloquium Pattern Recognition for Binary Images,* pp. 3/1-3/7, 7, 1989.

[9] Dong-Shik Lee and Young-Mo Kim, "Efficient Coding of Motion Vector and Mode Information for H.264/AVC," *Signal Processing: Image Communication,* Vol.24, Issue 10, pp. 834-839, 2009.

[10] 심동규, 조현호, 남정학, "HEVC (High Efficiency Video Coding) 최신 표준화 동향," 한국멀티미디어학회 학회지, Vol.14, No.2, pp. 1-15, 2010.

[11] 이동식, 김영모, "H.264/AVC에서 효율적인 움직임 벡터와 모드 정보의 압축," 한국멀티미디어학회 논문지, Vol.11, No.10, pp. 1359-1365, 2010.

이 동 식

1996년 경북대학교 (공학사 - 전자공학)
2000년 경북대학교 (공학석사 - 전자공학)
2010년 경북대학교 (공학박사 - 전자공학)
2011년 경북대학교 박사후과정
관심 분야 : 영상처리, 영상압축, 영상전송

김 영 모

1980년 경북대학교 (공학사 - 전자공학)
1983년 한국과학기술원 (공학석사 - 전자공학)
1989년 한국과학기술원 (공학박사 - 전자공학)

1997년 현재 경북대학교 공과대학 교수
관심 분야 : 컴퓨터 그래픽스, 멀티미디어, 비쥬얼 컴퓨팅, 영상처리