EVC 의 블록 분할 방식

박민수 *박민우 **최기호 ***표인지 ****최광표
삼성전자
mss.park@samsung.com *m.w.park@samsung.com **kiho14.choi@samsung.com
***yj1003.piao@samsung.com ****kp5.choi@samsung.com

Block partitioning in EVC

Minsoo Park *Min Woo Park **Kiho Choi ***Yinji Piao ****Kwang Pyo Choi
Samsung Electronics

요 약

본 논문에서는 차세대 비디오 압축 표준인 MPEG-5 Essential Video Coding (EVC) 에서 사용된 블록 분할 방식에 대해서 소개한다. EVC 에서 사용된 블록 분할 방식은 기존 비디오 압축 표준인 HEVC/H.265 에서 사용된 쿼드 트리(Quad-tree)가 아닌 이진 분할(Binary split)과 삼진 분할(Ternary split)을 사용한 Binary ternary tree(BTT) 기술을 사용하고 있다. 또한 기존 비디오 압축 기술과 달리 분할된 블록의 코딩 순서를 정해서 사용 할 수 있는 Split unit coding order (SUCO) 기술이 사용되고 있다.

1. 서론

비디오 압축 표준은 1996 년 제정된 MPEG-2 [1]에서부터, 2003 년 제정된 AVC/H.264 [2], 2013 년 제정된 HEVC/H.265 [3] 까지 이전 기술의 압축 성능을 2 배 향상시키면서 많은 기술의 발전이 있었다.

하지만 모든 비디오 압축 기술에서 공통적으로 사용하는 것은 블록 기반 코딩 방법이다. 한 장의 영상을 분할하지 않고 한번에 처리 하기에는 데이터의 용량이 크기 때문에 영상을 블록 단위로 분할하여 처리 한다.

MPEG-2, AVC/H.264 는 8x8 또는 16x16 매크로 블록 기반의 코딩 방법을 사용한다. HEVC/H.265 에서는 쿼드 트리(Quda-tree) 기반의 블록 분할 방식을 사용함으로써 고정된 크기가 아닌 64x64 부터 32x32, 16x16, 8x8, 4x4 까지 다양한 크기의 블록을 사용하고 있다.

HEVC/H.265 는 쿼드 트리 기반의 블록 분할 방식으로 다양한 크기의 블록을 사용함으로써 압축 성능을 크게 향상 시켰다. 하지만 블록의 모양이 정사각형에서 벗어나지는 못하였고 분할된 블록은 정해진 순서로 코딩이 되었다.

본 논문에서는 이진 분할(Binary split)과 삼진 분할(Ternary split) 방식을 사용해 정사각형이 아닌 다양한 모양의 블록을 사용하는 MPEG-5 Essential Video Coding(EVC) [4] 의 블록 분할 방식 인 Binary ternary tree(BTT) 및 분할 유닛의 코딩 순서를 정할 수 있는 Split unit coding order(SUCO) 기술을 소개함으로써 차세대 비디오 압축 기술의 발전 방향을 생각해 보도록 하겠다.

본 논문의 이후 구성은 다음과 같다. 2 절에서는 BTT 기술을 소개하고 3 절에서는 SUCO 기술을 소개한다. 4 절에서는 각 기술의 실험 결과를 살펴보고 5 절에서는 결론을 맺는다.

2. BTT (Binary and Ternary tree)

BTT 는 이진 분할 방식과 삼진 분할 방식을 조합하여 사용하는 블록 분할 방식이다.

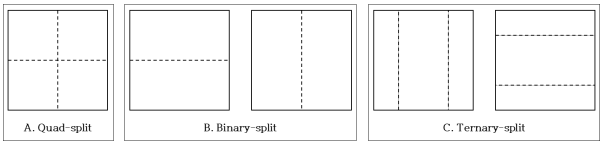

그림 1. 블록 분할 방식

위 그림 1-A 에서 보는 것과 같이 쿼드 분할 방식을 사용하면 정사각형 모양의 블록 밖에 사용할 수 없다. 하지만 그림 1-B 와 같이 이진 분할 방식을 사용하게 되면 정사각형 블록이 직사각형 모양으로 분할 되어 정사각형이 아닌 블록 모양을 사용 할 수 있게 된다.

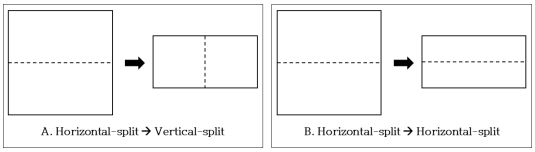

그림 2. 블록 이진 분할 방식

또한 이진 분할 방식을 그림 2-A 와 같이 서로 다른 방향으로 적용하면 쿼드 분할 방식과 동일하게 정사각형 블록으로 분할 할 수 있다. 그림 2-B 와 같이 서로 같은 방향으로 두 번 분할 하게 되면 더 긴 모양의 직사각형 블록을 사용할 수 있다.

이와 같이 이진 트리 방식을 사용하면 다양한 모양의 블록을 사용할 수 있지만 그에 따른 복잡도 증가는 문제가 될 수 있다. 예를 들어 쿼드 분할 방식의 경우 현재 블록에서 분할을 할지 안 할 지의 두 가지 선택이 가능한 반면 이진 분할 방식의 경우 분할을 할 경우 수평 방향인지 수직방향인지 혹은 분할하지 않을 것인지의 세가지 선택이 가능하다. 또한 쿼드 분할의 경우 현재 블록의 크기 2Nx2N 이 분할되면 NxN 4 개의 블록으로 분할이 되지만 이진 분할의 경우 2Nx2N 에서 Nx2N 또는 2NxN 으로 분할되기 때문에 쿼드 블록과 같은 NxN 블록이 되기 위해서는 두번의 분할이 필요하다.

이러한 선택이 한 단계에서 끝나는 것이 아닌 블록의 크기가 최소 블록이 될 때까지 반복적으로 수행되기 때문에 그 복잡도 차이는 단계를 거듭할수록 더욱 늘어나게 된다. EVC Main profile 의 BTT 에서는 이러한 복잡도를 줄이면서 다양한 블록을 활용하기 위해 1:1 모양의 정사각형 블록부터 1:4(4:1) 모양의 블록까지 사용하고 있다.

그림 1-C 의 삼진 분할의 경우 분할 되는 변의 비율이 1:2:1 이 되게 분할을 하고 있다. 쿼드 분할 및 이진 분할에서는 새로 분할된 블록의 위치는 분할 되기 전 블록의 1/2 위치에 자리할 수 밖에 없었지만 1:2:1 삼진 분할의 경우 1/4 위치에도 블록이 자리할 수 있게 되었다.

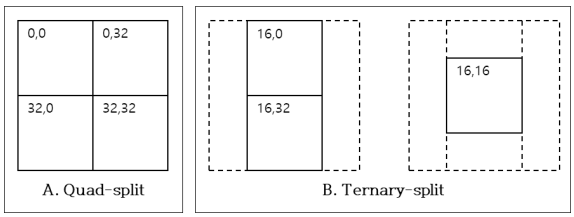

그림 3. 블록 분할 방식에 따른 블록 위치

예를 들어 그림 3-A 와 같이 32x32 블록의 경우 쿼드 트리에서는 블록의 왼쪽 위 샘플의 좌표 (x, y)가 32 의 배수가 되는 곳에만 자리할 수 있었다. 하지만 그림 3-B 에서 보는 것과 같이 삼진 분할을 사용하게 되면 32x32 블록의 좌표 (x, y)도 32 의 배수가 아닌 16 의 배수 위치에 자리 할 수 있게 된다. 이를 통해 기존 쿼드 블록으로는 표현 할 수 없는 위치의 블록까지 표현하게 되면서 더 다양한 형태의 블록 분할 모양을 사용할 수 있게 되었다.

이와 같이 이진 블록 분할 방식으로 쿼드 분할 방식 대비 더 다양한 모양의 블록을 사용할 수 있게 되었고 삼진 블록 분할 방식 사용으로 쿼드 분할 방식 대비 더 다양한 위치에 블록을 가질 수 있게 되었다. 이를 그림으로 보면 그림 4 와 같다.

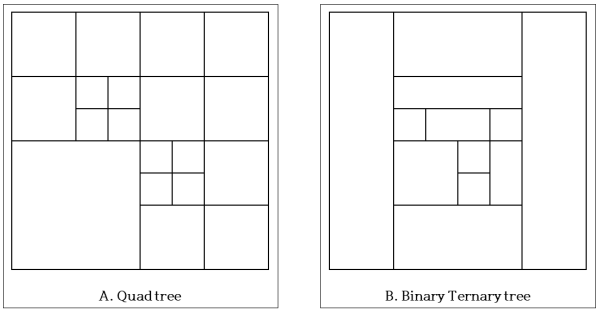

그림 4. 쿼드 트리 및 BTT

BTT 는 그림 4-B 에서 보는 바와 같이 이진 분할과 삼진 분할을 조합해서 사용하는 분할 방식이다. 이진 분할 후 삼진 분할을 사용할 수도 삼진 분할 후 이진 분할을 사용할 수도 있다. 이를 통해 쿼드 분할로만 이루어진 그림 4-A 의 블록 분할 모양을 만들 수도 있다. 또한 그림 4-B 의 블록 분할 모양과 같이 쿼드 분할로는 만들 수 없는 모양의 블록을 쿼드 분할로는 자리 할 수 없는 위치에 자리할 수 있게 함으로써 쿼드 트리 대비 더 많은 형태의 블록 분할 모양을 가지고 영상에 최적화된 블록 분할 모드를 찾음으로써 코딩 성능 향상을 기대할 수 있다.

3. SUCO (Split unit coding order)

SUCO 는 분할 유닛(Split unit, SU)에서 분할된 블록의 코딩 순서를 결정해 주는 기술이다. 여기서 분할 유닛이란 현재 블록이 더 작은 블록으로 분할 할 때 분할 되기 전 블록을 가리킨다.

분할 유닛은 쿼드 분할, 이진 분할, 삼진 분할 등으로 분할 될 수 있는데 그림 5 와 같이 기존 분할 방식에서는 정해진 순서에 따라 분할된 블록이 코딩 된다.

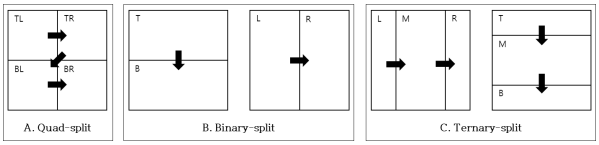

그림 5. 여러 블록 분할 방식에서 코딩 순서

하지만 블록의 코딩 순서는 다양한 조합으로 설정 가능하다. 예를 들어 쿼드 분할의 경우 그림 5-A 의 TL→TR→BL→BR 순서 외에도 BR→BL→TR→TL 등 24 가지 조합이 가능하다. 하지만 2 절에서 본 것 과 같이 블록 분할 방식이 다양해 짐으로써 그 복잡도가 증가 하였는데 순서의 다양한 조합을 블록 분할 방식과 조합하게 되면 그 복잡도는 블록 분할 복잡도와 블록 순서 복잡도의 곱으로 증가하게 된다.

EVC Main profile 의 SUCO 는 최소한의 복잡도 증가와 코딩 순서 변경의 이점을 얻기 위해 그림 6 과 같이 왼쪽에서 오른 쪽 방향 혹은 오른쪽 에서 왼쪽 방향의 코딩 순서를 선택적으로 사용할 수 있게 한다.

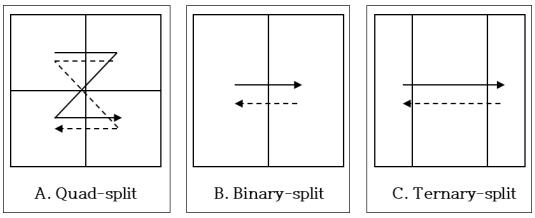

그림 6. SUCO 블록 코딩 순서

또한 분할 유닛의 상위 단계에서 코딩 순서가 오른쪽에서 왼쪽 방향으로 변경되었다면 변경된 이후의 하위 단계에서는 상위 단계의 변경된 방향을 따르게 되어있다.

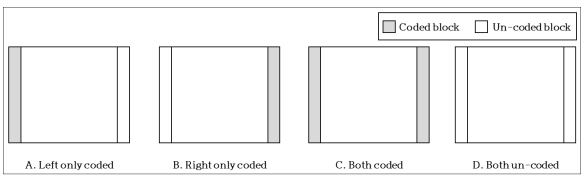

그림 7. 사용 가능한 주변 블록 정보

이와 같이 블록의 코딩 순서가 변경이 되면 그림 7 과 같이 사용 가능한 주변 블록의 정보도 변경이 된다. 그림 7-A 는 왼쪽 블록의 정보만 사용 가능한 경우로 블록 코딩 순서가 왼쪽→오른쪽 인 경우에 나올 수 있다. 그림 7-B 는 오른쪽 블록만 사용 가능한 경우로 그림 6-C 의 삼진 분할에서 코딩 순서가 오른쪽→왼쪽인 경우 가운데 블록을 코딩할 때 나올 수 있다. 그림 7-C 는 양쪽 모두 코딩 된 경우로 상위 분할 유닛의 코딩 순서는 왼쪽→오른쪽 이지만 현재 블록의 코딩 순서는 오른쪽→왼쪽일 때 마지막 코딩 순서인 블록에서 나올 수 있다. 그림 7-D 는 양쪽 모두 코딩이 되지 않은 경우로 코딩 순서가 왼쪽→오른쪽인 경우 영상의 첫번째 블록인 경우 혹은 코딩 순서가 오른쪽→왼쪽일 때 첫번째 분할 블록인 경우가 될 수 있다.

이렇게 SUCO 는 코딩 순서를 변경하는 것뿐만 아니라 이를 통한 주변 정보의 활용 가능성도 변경함으로써 코딩 성능 향상을 기대할 수 있다.

4. 실험 결과

앞에서 살펴본 BTT 와 SUCO 의 성능을 다음과 같은 실험을 통해서 알아 보았다. 실험은 EVC 의 Common test condition(CTC) [5] 에서 사용하는 테스트 영상을 기준으로 EVC reference software 인 EVC test model(ETM) 6.1 [6] 버전을 이용하여 실험을 진행하였다.

Test Sequence		BD-rate(Y)	BD-rate(U)	BD-rate(V)	EncTime	DecTime
4K	Tango2	-17.31%	-20.89%	-20.26%	555.24%	79.67%
	FoodMarket4	-16.68%	-18.53%	-18.59%	304.38%	93.62%
	CatRobot1	-19.80%	-28.86%	-25.73%	642.42%	83.55%
	DaylightRoad2	-23.23%	-29.56%	-27.46%	659.75%	81.16%
	ParkRunning3	-9.92%	-11.86%	-13.47%	872.88%	92.01%
2K	MarketPlace	-17.09%	-20.77%	-21.12%	619.39%	103.78%
	RitualDance	-20.86%	-26.35%	-25.94%	652.21%	116.23%
	Cactus	-16.19%	-23.72%	-21.28%	845.75%	94.53%
	BasketballDrive	-22.68%	-31.39%	-30.84%	691.75%	105.35%
	BQTerrace	-15.67%	-20.06%	-22.56%	759.19%	117.42%
	Average	**-17.94%**	**-23.20%**	**-22.73%**	**660.29%**	**96.73%**

표 1. BTT Test 결과

Test Sequence		BD-rate(Y)	BD-rate(U)	BD-rate(V)	EncTime	DecTime
4K	Tango2	-0.92%	-1.68%	-1.33%	156.03%	101.51%
	FoodMarket4	-0.23%	-0.12%	-0.48%	140.65%	94.99%
	CatRobot1	-1.17%	-2.49%	-2.25%	141.49%	98.61%
	DaylightRoad2	-1.06%	-1.86%	-1.30%	153.92%	108.37%
	ParkRunning3	-0.42%	-0.62%	-0.78%	143.22%	98.90%
2K	MarketPlace	-0.47%	-0.47%	-1.06%	154.90%	99.84%
	RitualDance	-1.13%	-1.80%	-1.28%	152.49%	98.66%
	Cactus	-0.56%	-1.95%	-1.96%	144.77%	95.52%
	BasketballDrive	-1.74%	-1.66%	-2.06%	151.05%	99.74%
	BQTerrace	-0.64%	-1.17%	-1.99%	160.60%	124.02%
	Average	**-0.83%**	**-1.38%**	**-1.45%**	**149.91%**	**102.02%**

표 2. SUCO Test 결과

표 1, 표 2 의 결과는 EVC Main profile 에서 대부분의 기술을 끈 상태에서 각각의 툴을 사용하였을 경우의 결과이다.

EVC Main profile 에서 BTT 를 사용하지 않을 경우 EVC Baseline profile 의 블록 분할 방식인 쿼드 트리 방식을 사용하게 된다.

표 1 의 결과를 보면 EVC Main profile 의 BTT 기술은 쿼드 트리 방식의 기술 대비 약 18%의 BD-rate(Y) 성능 향상을 약 6.6 배의 인코더 복잡도로 가져온다. 이는 2 절에서 알아본 바와 같이 쿼드 트리 대비 BTT 의 복잡도가 높지만 다양한 분할 방식을 통해 압축 성능을 향상시킬 수 있다는 것을 보여준다.

표 2 의 결과를 보면 EVC Main profile 의 SUCO 기술은 쿼드 트리 방식에서 고정된 순서로 코딩 하는 것 대비 약 0.8%의 BD-rate(Y) 성능 향상을 약 1.5 배의 인코더 복잡도로 가져온다. 이는 3 절에서 알아본 바와 같이 다양한 순서로 코딩을 함으로써 복잡도는 높아지지만 코딩 성능을 향상 시킬 수 있다는 것을 보여준다.

5. 결론

MPEG-5 Essential Video Coding 에서는 기존 영상 압축 기술과 달리 더 다양한 모양의 블록을 사용함으로써 약 18%의 압축 성능 향상을 가져왔고 다양한 순서로 분할된 블록을 코딩함으로써 약 1%의 압축 성능 향상을 가져왔다. 하지만 다양한 블록 분할 방식은 약 6.6 배, 다양한 순서의 코딩은 약 1.5 배의 인코더 복잡도 향상도 같이 가져왔다. EVC 에서는 복잡도와 코딩 성능 사이에서 가장 효율적인 방식으로 현재 EVC 의 블록 분할 방식을 채택하게 되었다. 앞으로 더 다양한 모양 및 순서의 블록 분할 방식을 사용하면서도 어떤 분할 방식을 사용할지에 대한 선택을 빠르게 하는 방법을 찾는다면 영상 압축 기술에서 새로운 블록 분할 방식을 통해 더 큰 성능 향상을 기대해 볼 수 있을 것이다.

참조

[1] Generic coding of moving pictures and associated audio information—Part 2: Video, ISO/IEC 13818 2 MPEG-2 and ITU-T Recommendation H.262 (1994).

[2] Advanced Video Coding (AVC), ITU-T Recommendation H.264 and ISO/IEC 14496-10 (2003).

[3] High Efficient Video Coding (HEVC), ITU-T Recommendation H.265 and ISO/IEC 23008-2 (2013).

[4] Text of ISO/IEC FDIS 23094-1 Essential Video Coding, ISO/IEC JCT 1/SC 29/WG 11 N 19229 (2020)

[5] Common Test Conditions for Essential Video Coding, ISO/IEC JCT 1/SC 29/WG 11 N19003 (2020).

[6] EVC Reference Software version 6.1. [Online]. Available: https://gitlab.com/MPEG-5/ETM/-/tree/ETM6.1