Journal of Korea Multimedia Society Vol. 21, No. 1, January 2018(pp. 18-25)
https://doi.org/10.9717/kmms.2018.21.1.018

상위 블록 움직임 벡터를 이용한 HEVC 움직임 예측 탐색 범위 감소 기법

이 규 중†

Search Range Reduction Algorithm with Motion Vectors of Upper Blocks for HEVC

Kyujoong Lee†

ABSTRACT

In High Efficiency Video Coding (HEVC), integer motion estimation (IME) requires a large amount of computational complexity because HEVC adopts the high flexible and hierarchical coding structures. In order to reduce the computational complexity of IME, this paper proposes the search range reduction algorithm, which takes advantage of motion vectors similarity between different layers. It needs only a few modification for HEVC reference software. Based on the experimental results, the proposed algorithm reduces the processing time of IME by 28.1% on average, whereas its the Bjøntegaard delta bitrate (BD-BR) increase is 0.15% which is negligible.

Key words: Search Range Reduction, Integer Motion Estimation, Computational Complexity, Upper Motion Vectors

1. 서 론

High Efficiency Video Coding (HEVC) [1]는 H.264/AVC의 뒤를 이어 조인트 비디오 팀에 의해 2013년도에 제안된 영상 압축 표준이다. HEVC는 여러 새로운 기술을 채택함으로써 압축 성능을 H.264/AVC 대비 2배로 향상시켰다. 개선된 압축 성능을 기반으로, HEVC는 고해상도 및 고화질 영상이 요구되는 분야에서 널리 활용되고 있다.

HEVC는 압축 성능 개선을 위해 매우 유연한 계층적 블록 구조를 채택하였다[2]. HEVC는 화면을 정사각형의 Coding Tree Unit (CTU) 단위로 구분하고, 각 CTU는 쿼드 트리 형태로 재귀적으로 분할된다. 분할되는 단위를 Coding Unit(CU)로 정의하며, 각 CU는 2N×2N, N×2N, 2N×N, N×N인 대칭 구조와 비대칭 구조의 Prediction Unit (PU) 단위로 분할된다. 각각의 PU에 대해서 HEVC 부호화기는 움직임 예측을 통해 최적의 움직임 벡터를 탐색하여, 최적의 분할 구조를 결정한다. CTU는 기본적으로 4개의 레이어로 재귀적으로 분할이 가능하기 때문에, 이러한 복잡한 블록 구조는 HEVC 부호화기의 연산 복잡도를 크게 증가시켰고, 또한 중복되는 움직임 정보로 인해 정보량의 증가도 야기하였다[3]. 특히, 움직임 예측은 각각의 PU에 대해 최적의 움직임 벡터를 탐색하기 때문에, 연산의 복잡도가 매우 높은 편이다. 탐색 범위를 이용하는 정수 단위 움직임 예측은 HEVC

※ Corresponding Author : Kyujoong Lee, Address: (31460) 70, Sunmoon-ro 221 beon-gil, Tangjeong-myeon, Asan-si, Chungcheongnam-do, Korea, TEL : +82-41-530-2271, FAX : +82-41-530-2933, E-mail : kyujoonglee@sunmoon.ac.kr
Receipt date : Nov. 3, 2017, Revision date : Dec. 21, 2017
Approval date : Dec. 28, 2017
† Dept. of Electronic Eng., School of Engineering, Sun Moon University
※ This research was supported by the Sun Moon University Research Grant of 2017.

전체 부호화기 연산 복잡도의 40% 정도를 차지하는 것으로 알려져 있다[4]. 이러한 정수 단위 움직임 예측을 줄이기 위해 여러 가지의 연구가 진행되어 왔다. 우선, 움직임 예측의 탐색 시작점을 정확하게 찾아서 비트율을 최소화 하면서 매우 적은 탐색 포인트들만 탐색하는 방향의 연구가 있었다[5,6]. 여러 효율적인 탐색 알고리즘들에 대한 연구가 있었고, HEVC는 TZS(Test Zone Search) 알고리즘을 채택하였다. 이후 TZS를 개선하는 연구들이 진행되었다[7-9].

이와 다른 방향의 연구로, 최적 모드를 조기에 예측하여 움직임 예측 연산량을 줄이는 기법들인 ECU (Early CU Termination)[10], ESD (Early Skip Detection)[11], CFM (Coded Block Flag Fast Method)[12]이 제안되었고, HEVC에 채택되었다. ECU는 해당 CU 분할이 더 이상 필요 없는지를 판단하여 연산량을 줄이는 기술이고, ESD는 Skip 모드 조건을 만족하면 더 이상의 PU 분할을 하지 않아 연산량을 줄인다. CFM은 ESD가 활성화된 상태에 적용되는 기술로, PU에 대한 CBF(Coded Block Flag)가 0이면 더 이상 PU 분할을 하지 않는다. [13]에서는 CU와 PU의 분할을 중단하는 조건과 임계치 설정 등을 수정하여 좀 더 연산량을 줄이는 방법을 제안하고 있다. 이러한 최적 모드 조기 예측 방법 등을 적용한다 해도, 생략되지 않는 CU와 PU에 대해 여전히 움직임 예측은 수행되어야하기 때문에, 효과적인 탐색 범위 감소 기법을 적용하면 압축 성능 저하를 최소화 하면서 추가로 연산량을 줄일 수 있다.

본 논문에서 상위 레이어의 움직임 벡터를 이용하여, 하위 레이어의 움직임 예측 탐색 범위를 감소시키는 기법을 제안한다. H.264/AVC에서도 다른 레이어와의 관계를 이용하여 움직임 예측 연산을 효과적으로 줄이는 연구들[14-16]이 있었으나, HEVC에 비해 블록 구조가 단순하기 때문에 연산시간 감소에 제한이 있다. [17]에서는 2Nx2N의 움직임 벡터를 이용하여, 같은 레이어의 다른 PU에 대한 탐색 범위를 줄여서 연산량을 감소시키는 방법을 제안하고 있다. 2Nx2N에 대한 연산이 항상 수행되어야 하며, PU 분할이 반영이 되지 않기 때문에, 본 논문에서 제안된 방법에 비해 연산 감소량이 약간 작은 편이다. [18]은 상위 레이어와 하위 레이어 간의 움직임 벡터가 유사함을 이용하여, 본 논문 제안과 반대로 하위 레이어의 움직임 벡터를 이용하여 상위 레이어의 움직임 벡터를 예측함으로써 연산량을 줄이는 방법을 제안하고 있다. HEVC 참조 소프트웨어는 상위 레이어의 CU에서 하위 레이어의 CU를 재귀 호출하는 순서로 진행되기 때문에, 하드웨어 구현은 용이할 수 있지만 소프트웨어 구현이 쉽지 않다. 또한, 연산량을 크게 줄일 수 있는 최적 모드 조기 예측 방법들은 상위 레이어서 하위 레이어로 진행되기 때문에 두 방법을 조합하여 사용하기에 한계가 있다. 반면, 제안하는 탐색 범위 감소 기법은 기존 CU 처리 순서와 동일하여, 기존 방법들과 조합하여 사용하기가 용이하다. 제안하는 기법은 서로 다른 레이어의 동일 위치의 최적 움직임 벡터가 유사하다는 특징을 기반으로 압축 성능 저하를 최소화 하면서 움직임 예측 연산 시간을 평균적으로 28.1% 줄일 수 있고, 최적 모드 조기 예측 방법과 조합하여 추가 연산 감소가 가능하다.

본 논문은 총 네 개의 장으로 이루어지며 2장부터의 구성은 다음과 같다. 2장에서는 정수 단위 움직임 예측에 대해 설명하고, 상위 레이어와 하위 레이어의 움직임 벡터가 유사하다는 실험 자료를 기반으로 제안된 탐색 범위 감소 기법에 대해서 설명한다. 3장에서는 제안하는 알고리즘을 구현한 HM13.0[19]을 이용하여 측정한 압축 성능 저하와 연산 시간 감소와 최적 모드 조기 예측 방법과 조합 등의 실험 결과를 제시하고 분석하며, 4장에서는 본 논문의 결론을 기술한다.

2. 제안하는 움직임 예측 탐색 범위 감소 기법

제안하는 기법의 이해를 돕기 위해, HEVC의 CU 처리 과정과 움직임 예측 설명을 먼저 하도록 한다. HEVC의 움직임 예측은 Fig. 1에서 점선에 해당되는 PU 들에 대해서 실행된다. CU는 Merge 모드를 먼저 처리한 후, 2N×2N, N×2N, 2N×N, N×N 순서로 대칭 모드를 처리한 후 비대칭 모드를 처리한다. Fig. 1의 AMP (Asymmetric Motion Partition)는 비대칭 모드를 의미하고, N×N은 CU의 깊이가 제일 깊을 때만 수행된다. 각각의 PU를 처리한다는 것은 정수 단위 움직임 예측, 분수 단위 움직임 예측을 한 후 정해진 최적 움직임 벡터를 이용하여 잔차 신호에 대해 RD (Rate-Distortion) Cost를 계산하여 Cost가 가장 작은 최적 PU를 결정하는 것을 의미한다. 본 논문에서 제안하는 탐색 범위 감소 기법은 정수 단위 움직임

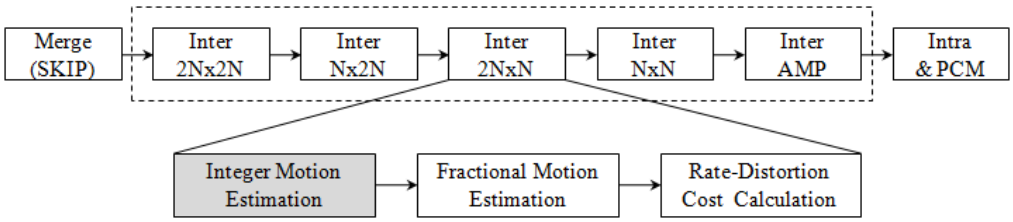

Fig. 1. Encoding flow for CU.

예측의 탐색 범위 설정에 적용되어 해당 연산량을 감소시킨다.

정수 단위 움직임 예측은 Fig. 2에 나타내는 것처럼 움직임 예측을 적용할 현재 PU에 대한 최적의 PMV (Predicted Motion Vector)를 결정한 후, 결정된 PMV를 중심으로 참조 프레임의 탐색 범위 내에서 움직임 예측을 수행하여 SAD(Sum of Absolute Difference) 기반의 RD Cost가 최소인 최적 움직임 벡터를 정한다. HEVC에서는 PMV의 성능을 향상하기 위해 AMVP(Advanced Motion Vector Prediction)[19]를 채택하여, 공간에서 이웃하는 PU로부터 5개의 후보들과 시간 축에서 동일 위치의 PU로부터 2개의 후보들 총 7개 후보에서, 우선 순위가 높은 2개의 PMV 후보를 선정한다. 정수 단위 움직임 예측 시작시 2개의 PMV 후보들 중에 RD cost가 작은 쪽으로 PMV를 최종 선택한다. 이렇게 구해진 PMV를 중심으로 둔 ±64 탐색 범위를 TZS 탐색 알고리즘이 수행되면서 최적 정수 단위 움직임 벡터를 결정한다. 탐색된 최적 움직임 벡터를 기준으로 분수 단위 움직임 예측이 수행되고, 최종적으로 결정된 움직임 벡터와 PMV의 차이가 엔트로피 코더에 의해 부호화 된다. 그 차이가 커질수록 움직임 벡터에 대한 부호화된 비트수는 증가하게 된다.

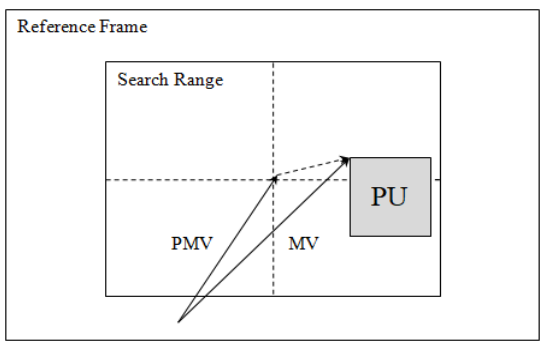

Fig. 2. Integer motion estimation.

HEVC 참조 소프트웨어인 HM13.0 [19]의 부호화기의 실행 순서를 분석해 보면, 상위 레이어의 CU에서 하위 레이어의 CU를 재귀 호출하는 순서로 진행된다. 따라서, 상위 CU의 PU에 대해서 움직임 예측이 완료된 이후에, 대응되는 하위 CU의 PU에 대해서 움직임 예측이 수행되는 구조이다. 이러 이유로, 하위 CU에 대해 움직임 예측을 할 때, 대응되는 상위 CU의 PU에 대한 최적의 움직임 벡터는 사용이 가능하다. 상위 CU의 PU에 대한 움직임 벡터와 대응되는 하위 CU의 PU의 움직임 벡터가 유사할 가능성이 높기 때문에, 본 논문에서는 이를 정량적으로 분석하여, 이를 기반으로 하위 CU의 PU에 대한 움직임 예측 탐색 범위를 줄이고자 한다.

CU 레이어간의 움직임 벡터의 유사의 정도를 정량적으로 분석하기 위해, 실험을 진행하여 그 결과를 Table 1에 정리하였다. 앞서 설명한 것처럼, HEVC는 총 4개의 CU 레이어를 지원한다. 이를 깊이로 표현하며 최상위 레이어는 깊이가 0이고 최하위 레이어는 깊이가 3이 된다. 각 CU 레이어의 모든 PU에 대한 움직임 예측과 RD cost 비교가 종료되면 최적 PU 분할과 최적 움직임 벡터가 결정된다. 이러한 최적 움직임 벡터를 뽑아서, Fig. 3과 같이 다른 깊이 동일한 위치의 움직임 벡터의 차를 구해 평균을 구하였다. 계산의 편의를 위해 움직임 벡터의 차는 맨해튼 거리로 계산하여, 예를 들면 Fig. 3의 (5,6)과 (4,8)의 차이는 3으로 구해진다. Table 1은 Class A인 영상 “Traffic”, “PeopleOnStreet”와 Class B인 영상 ”BQTerrace”, “BasketballDrive“의 통계를 나타내고 있다. 정적인 영상인 “Traffic”의 경우 깊이가 0인 레이어의 움직임 벡터와 깊이가 1인 레이어의 움직임 벡터와의 차이는 0.64 로 매우 작다. 더 나아가 깊이가 3인 레이어의 움직임 벡터와의 차이도 0.80으로 여전히 1 픽셀보다도 작음을 알 수 있다. 동적인 영상인 “BasketballDrive”의 경우 깊이가 0인 레이어

Table 1. Average Difference of Motion Vectors Between Different Depths for “Traffic”, “PeopleOnStreet”, “BQTerrace” and “BasketballDrive”

	Average Difference of Motion Vectors Between Depths					
Sequences	Traffic			PeopleOnStreet		
Depth	1	2	3	1	2	3
0	0.64	0.78	0.80	2.45	2.93	2.79
1		0.63	0.70		2.69	2.80
2			0.54			2.32
Sequences	BQTerrace			BasketballDrive		
Depth	1	2	3	1	2	3
0	1.79	2.11	2.17	5.03	5.72	5.61
1		1.80	2.04		5.08	5.55
2			1.29			3.96

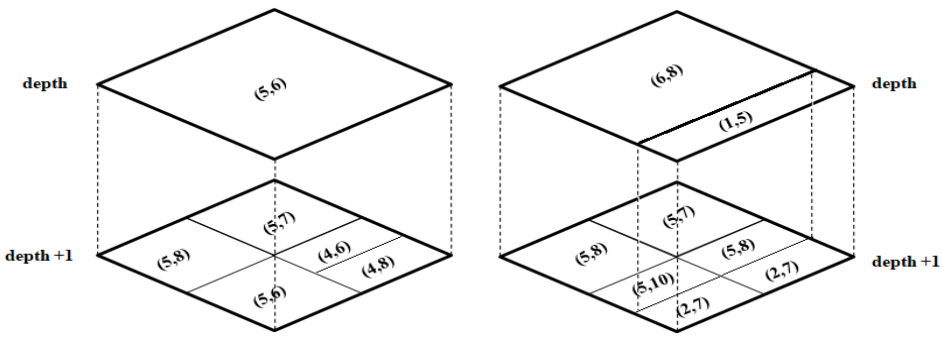

Fig. 3. Examples of best motion vectors between different depths.

의 움직임 벡터와 깊이가 1인 레이어의 움직임 벡터와의 차이가 5.03으로 증가하지만 그 차이가 크지 않다. 또한, 깊이의 차이가 벌어져서 움직임 벡터의 차이가 늘어나도 여전히 6픽셀 미만으로 크지 않다. 이러한 경향은 다른 영상들에서도 비슷하게 나타난다.

제안하는 탐색 범위 감소 기법은 최상위 레이어의 모든 움직임 예측이 종료된 이후 채택된 최적 움직임 벡터를 이용하여, 이후 처리되는 하위 레이어들의 움직임 예측 탐색 범위를 감소시키는 방법이다. Table 1의 결과처럼 최상위 레이어의 움직임 벡터와의 차이와 근접한 상위 레이어의 움직임 벡터와의 차이가 크지 않기 때문에, 구현의 단순화를 위해 최상위 레이어의 최적 움직임 벡터만을 활용한다.

$$SR = \begin{cases} 64 & \text{for } depth = 0 \\ \min(64, MVD_{MAX} + d) & \text{for } depth > 0 \end{cases} \quad (1)$$

$$MVD_{MAX} = \max(|MVX_{UPPER} - PMVX|, |MVY_{UPPER} - PMVY|) \quad (2)$$

제안하는 방법은 식(1)로 표현된다. “depth=0”을 만족하는 최상위 레이어의 경우 기존과 같이 PMV를 시작점으로 ±64의 영역을 탐색하게 되고, “depth>0”을 만족하는 레이어의 경우, 식(2)를 통해 MVD_{MAX}를 계산해야 한다. 최상위 블록 PU의 X 방향의 움직임 벡터(MVX_{UPPER})와 PMV의 X 방향의 움직임 벡터의 절대값의 차이와 최상위 블록 PU의 Y 방향의 움직임 벡터(MVY_{UPPER})와 PMV의 Y 방향의 움직임 벡터의 절대값의 차이 중 큰 값으로 MVD_{MAX} 정해진다. 압축 성능 저하를 최소화하기 위해 ±d 만큼 탐색 범위를 키우게 된다. 제안된 방법은 연산량을 줄이는 것이 목표이기 때문에, 계산된 최대 탐색 범위를 ±64로 제안하기 위해 min을 적용하였다. 3장의 실험 결과는 d를 4로 설정했을 때의 결과로 실험적으로 정하였다. Fig. 4은 제안된 방법이 재귀호출 방식의 CU 인코딩 방식에 구현되었을 때의 처리 순서를 나타내고 있다. “depth=0”일 때, 비대칭 PU까지 처리가 되면, 최상위 움직임 벡터가 확정된다. 이를 하위 레이

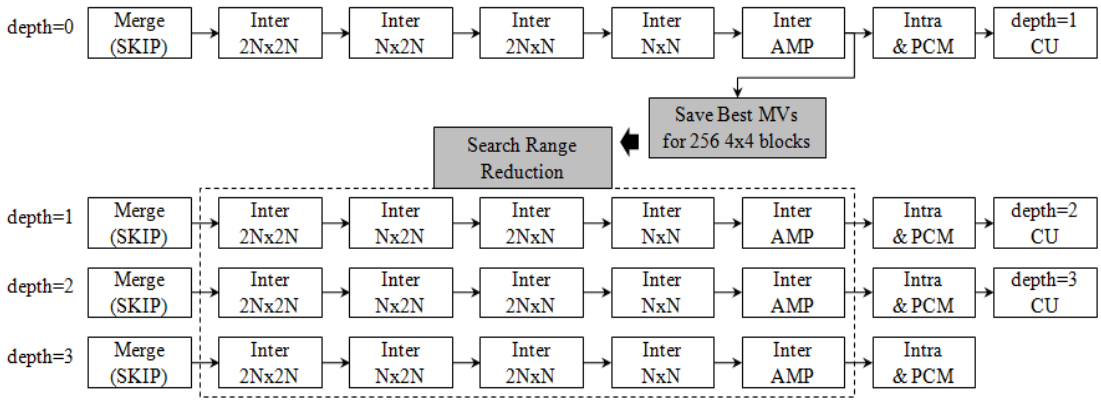

Fig. 4. Flow chart of proposed method in recursive CU encoding.

어에 사용하기 위해서 저장해야 하는데, 최하위 레이어의 경우 4×4 크기의 PU까지 존재하기 때문에, 64×64에 대한 움직임 벡터를 256개 4×4 블록 단위로 저장을 한다. 이후 하위 레이어의 PU의 움직임 예측이 시작될 때 제안된 방법이 적용된 축소된 탐색 범위에 대해서 움직임 예측이 수행된다. 기존 방법인 [16]를 구현하기 위해서는, 2N×2N의 움직임 벡터를 사용하고, 대칭 모드일 때는 d=4를, 비대칭 모들일 때는 d=8을 적용하면 된다.

3. 실험 결과 및 고찰

제안된 탐색 범위 감소 기법은 HM 13.0 [19]에 구현되었고, 압축 성능 저하 정도와 정수 단위 움직임 예측의 연산 시간 감소율이 측정되었다. 실험은 CentOS 7.3 운영체제, 3.5GHz의 Intel Xeon E3-1245V5 그리고 DDR4 64 GB의 사양을 갖춘 서버에서 수행되었다. 제안된 방법의 성능은 같은 레이어의 2N×2N PU의 최적 움직임 벡터를 활용한 방법[17]과 비교하였고, 최적 모드 조기 예측 방법들과 조합으로 인한 효과도 비교 분석하였다. Low Delay P 조건으로 Class A 영상 2개, Class B 영상 4개, Class C 영상 4개 그리고 움직임이 크고 복잡한 1920×1080 해상도 영상 3개를 추가하여, 총 13개의 영상을 실험하였고, 각 영상당 100개의 프레임을 인코딩하였다. 압축 효율은 동일 PSNR 대비 비트율의 변화를 나타내는 Bjøntegaard delta bit rate (BD-BR) [21]을 이용하여 측정하였고, 양자화 파라미터(Quantization Parameter:QP)는 22, 27, 32 그리고 37을 사용하였다. BD-BR 증가율이 양수인 경우 제안된 방법의 압축 효율이 저하되었음을 나타낸다. 정수 단위 움직임 예측의 연산 시간 감소량은 식(3)을 이용하여 측정하였다. TS(Time Saving)는 연산 시간의 감소량을 백분위로 나타낸 수치이고, T(Reference)는 기존 HM 13.0의 정수 단위의 움직임 예측 연산 시간을 T(Proposed)은 제안된 방법의 정수 단위의 움직임 예측 연산 시간을 나타낸다. 이때, 정수 단위의 움직임 예측 연산 시간은 모든 레이어에 대해서 측정하였다. 최적 모드 조기 예측 방법들과의 조합을 실험할 때는, 정수 단위의 움직임 예측 뿐만 아니라 분수 단위 움직임 예측 그리고 RD Cost 계산 등의 연산량이 모두 감소하기 때문에, T(Reference)와 T(Proposed)를 전체 인코딩 시간으로 측정하였다.

$$TS(\%) = (T(\text{Reference}) - T(\text{Proposed})) / T(\text{Reference}) \quad (3)$$

Table 2는 기존 HM13.0 대비 제안된 방법들에 의한 연산 시간 감소율과 압축 효율 저하에 대한 실험 결과를 제시하고 있다. “T_IME”는 정수 단위 움직임 예측의 실제 수행된 시간을 나타내고 있다. 제안된 방법의 정수 단위의 움직임 예측 연산 시간의 감소율은 평균 28.1% 인 반면에 BD-BR의 증가율은 0.15%로 매우 미미하다. 기존 방법인 [17]의 BD-BR의 증가율은 0.05% 으로 제안된 방법보다 작지만, 연산 시간 감소율은 또한 18.4% 로 작다. 2N×2N에 대한 연산을 항상 수행해야 하며, PU 분할이 적용되지 않기 때문에, 연산 시간 감소율이 제안된 방법보다 평균 9.7% 작다. 그러나, 제안된 방법은 상위 레이어의 움직임 벡터를 활용하고, 기존 방법은 같은 레이어의 움직임 벡터를 활용하기 때문에, 두 가지

Table 2. BD-BR Increase and Time Saving of the Previous and the Proposed Method

			Previous Method[16]		Proposed Method	
Class	Sequences	T_IME (sec)	BD-BR Inc(%)	Time Saving(%)	BD-BR Inc(%)	Time Saving(%)
Class A (2560×1600)	Traffic	725.7	0.11	18.0	0.14	29.3
	PeopleOnStreet	2761.3	0.19	25.1	0.57	38.3
Class B (1920×1080)	BasketballDrive	1423.7	0.14	8.3	0.09	14.6
	BQTerrace	504.2	-0.02	18.1	0.38	28.0
	Cactus	693	0.13	23.1	0.12	35.1
	ParkScene	598	-0.03	19.5	0.01	29.9
Class C (832×480)	BQMall	166.6	0.18	15.9	0.36	28.4
	BasketballDrill	165.3	0.13	19.8	0.34	33.0
	PartyScene	143.3	0.07	22.0	0.16	36.9
	RaceHorses	366.9	-0.24	7.5	-0.15	18.6
Large & Complex Motions (1920×1080)	PedestrianArea	1418.4	-0.18	22.3	0.07	24.0
	RushHour	1081.5	0.19	25.6	-0.08	30.3
	Tractor	1706.2	-0.05	14.4	0.00	18.4
	Average		0.05	18.4	0.15	28.1

방법을 최적의 방식으로 조합하면, 압축 성능 저하는 최소화 하면서 연산을 추가로 줄이는 것이 가능할 것으로 예상된다. 이 부분은 다음의 연구의 과제로 남겨두기로 한다. 대체로 움직임이 적은 영상에서의 연산량 감소의 폭이 크고, 움직임이 클수록 연산량 감속의 폭이 작다. 이러한 현상은 움직임이 큰 영상일수록 선택된 PMV와 최상위 레이어의 움직임 벡터의 차이가 날 확률이 증가하기 때문이다. 그러나, 움직임이 큰 영상인 “BasketballDrive”와 “RaceHorses”에서도 15% 이상의 연산 시간을 줄이면서, 압축 효율 저하는 거의 0에 가까운 수치를 보이고 있다. 움직임이 훨씬 크고 복잡한 영상의 조합인 “Large & Complex Motion” 그룹의 영상에서도, 18% 이상의 연산 시간을 줄이면서, 압축 효율 저하는 거의 0에 가깝다. 움직임이 복잡한 영상일 때, 압축 효율 저하가 미미한 이유는 다음과 같이 분석된다. 정수 단위 움직임 예측은 SAD 기반의 RD Cost를 기준으로 정해지지만, 최적 PU 분할과 CU의 깊이의 최종 결정은 엔트로피 코더에 의한 비트수와 SSE (Sum of Squared Error)의 합으로 구성된 RD Cost 기준으로 정해진다. 따라서, 복잡한 영상일수록 SAD 기반의 RD Cost로 정해진 최적 움직임 벡터를 SSE 기반의 RD Cost 기준으로 비교한다면 최적 움직임 벡터가 아닐 확률이 높아지게 된다. 이러한 이유로, 움직임이 복잡한 영상일 때, 최적 움직임 벡터가 바뀐다고 해도 그 정도가 크지 않으면 압축 성능에 미치는 영향이 미미하다.

Table 3은 HEVC에 채택된 최적 모드 조기 예측 기술인 ECU, ESD 그리고 CFM과 같이 사용할 때, 제안된 방법이 여전히 효과적임을 보여주고 있다. “T_ENC”는 전체 인코딩 시간을 나타낸다. Class B, C의 8개 영상에서 ECU+ESD+CFM의 조합은 전체 인코딩 시간을 26.7% 감소시켰고, 제안된 방법과의 조합으로는 6.7% 추가로 인코딩 시간을 줄였으며, 이때 BD-BR 증가는 0.15% 수준으로 미미하다.

4. 결 론

본 논문은 HEVC 부호화기에서 연산의 복잡도가 매우 높은 정수 단위 움직임 예측의 연산 시간을 줄이기 위해 탐색 범위를 감소시키는 기법을 제안하였다. 제안한 기법은 레이어간의 동일 위치 블록의 최적 움직임 벡터가 매우 유사하다는 점을 기반으로 하였다. 깊이가 1 이상인 레이어의 움직임 예측 연산을 할 경우, PMV을 중심으로 한 기존 탐색 범위 안에 최상위 레이어의 최적 움직임 벡터가 존재할 경우, 해당 위치까지 탐색 범위를 축소하여 연산 시간을 줄였다. 제안한 기법의 성능을 입증하기 위해, 연산 시간의 감소율과 압축 효율 저하를 고해상도 포함 13개의 테스트 영상에 대해 측정하였다. 제안한 탐색

Table 3. BD-BR Increase and Time Saving of the Fast Encoder Options and the Fast Encoder Options with Proposed Method

			ECU+ESD+CFM		ECU+ESD+CFM +Proposed Method	
Class	Sequences	T_ENC (sec)	BD-BR Inc(%)	Time Saving(%)	BD-BR Inc(%)	Time Saving(%)
Class B (1920×1080)	BasketballDrive	4155.7	0.73	27.8	0.82	32.6
	BQTerrace	3186	0.71	38.8	0.92	42.8
	Cactus	3184.3	1.47	32.0	1.55	39.9
	ParkScene	3121.4	1.01	38.8	1.06	43.9
Class C (832×480)	BQMall	691.7	1.11	26.1	1.4	33.0
	BasketballDrill	681.7	0.6	24.8	0.85	32.3
	PartyScene	798.6	0.9	16.3	0.96	23.4
	RaceHorses	1066.7	0.41	9.0	0.58	19.4
	Average		0.87	26.7	1.02	33.4

범위 감소 기법은 정수 단위 움직임 예측의 연산 시간을 평균적으로 28.1%로 감소시켰다. 반면에, 압축 효율 저하는 평균 0.15%의 비트율 증가 정도로 미미하다. 또한, HEVC에 채택된 ECU, ESD 그리고 CFM과 같은 최적 모드 조기 예측 기술과 함께 사용한 경우에도, 압축 효율 저하 거의 없이 추가로 연산량을 감소시켰다.

REFERENCE

[1] G.J. Sullivan, J.R. Ohm, W.J. Han, and T. Wiegand, "Overview of the High efficiency Video Coding (HEVC) Standard," *IEEE Transactions on Circuits and Systems for Video Technology*, Vol. 22, No. 12, pp. 1649-1668, 2012.

[2] I.K. Kim, J. Min, T. Lee, W.J. Han, and J. Park, "Block Partitioning Structure in the HEVC Standard," *IEEE Transactions on Circuits and Systems for Video Technology*, Vol. 22, No. 12, pp. 1697-1706, 2012.

[3] D.S. Lee and Y.M. Kim, "Efficient Motion Information Representation in Splitting Region of HEVC," *Journal of Korea Multimedia Society,* Vol. 15, No. 4, pp. 485-491, 2012.

[4] F. Bossen, B. Bross, K. Suhring, and D. Flynn, "HEVC Complexity and Implementation Analysis," *IEEE Transactions on Circuits and Systems for Video Technology*, Vol. 22, No. 12, pp. 1685-1696, 2012.

[5] S. Kamp and M. Wien, "Decoder-Side Motion Vector Derivation for Block-Based Video Coding," *IEEE Transactions on Circuits and Systems for Video Technology*, Vol. 22, No. 12, pp. 1732-1745, 2012.

[6] W.D. Chien, K.Y. Kiao, and J.F. Yang, "Enhanced AMVP Mechanism Based Adaptive Motion Search Range Decision Algorithm for Fast HEVC Coding," *Proceeding of IEEE International Conference on Image Processing*, pp. 3696-3699, 2014.

[7] N. Purnachand, L.N. Alves, and A. Navarro, "Improvements to TZ Search Motion Estimation Algorithm for Multiview Video Coding," *Proceeding of The International Conference on Systems, Signals and Image Processing*, pp. 388-391, 2012.

[8] H. Kibeya, F. Belghith, H. Loukil, M.A.B. Ayed, and N. Masmoudi, "TZSearch Pattern Search Improvement for HEVC Motion Estimation Modules," *Proceeding of International Conference on Advanced Technologies for Signal and Image Processing*, pp. 95-99, 2014.

[9] X. Li, R. Wang, W. Wang, Z. Wang, and S. Dong, "Fast Motion Estimation Methods for

HEVC," *Proceeding of IEEE International Symposium on Broadband Multimedia Systems and Broadcasting*, pp. 1-4, 2014.

[10] K. Choi, S.H. Park, and E.S. Jang, *Coding Tree Pruning Based CU Early Termination,* JCTVC-F092, 2011.

[11] J. Yang, J. Kim, K. Won, H. Lee, and B. Jeon, *Early Skip Detection for HEVC,* JCTVC-G543, 2011.

[12] R.H. Gweon, Y.L. Lee, and J. Lim, *Early Termination of CU Encoding to Reduce HEVC Complexity,* JCTVC-F045, 2011.

[13] K. Saurty, P.C. Catherine, and K.M.S. Soyjaudah, "Inter Prediction Complexity Reduction for HEVC Based on Residuals Characteristic," *The Journal of The Korean Institute of Communication Sciences*, Vol. 7, No. 10, pp. 310-318, 2016.

[14] K.C. Hou, M.J. Chen, and C.T. Hsu, "Fast Motion Estimation by Motion Vector Merging Procedure for H. 264," *Proceeding of IEEE International Conference on Multimedia and Expo*, pp. 1444-1447, 2005.

[15] Y.K. Tu, J.F. Yang, Y.N. Shen, and M.T. Sun, "Fast Variable-size Block Motion Estimation Using Merging Procedure with an Adaptive Threshold," *Proceeding of IEEE International Conference on Multimedia and Expo*, pp. 789-792, 2003.

[16] A. Chang, P.H.W. Wong, Y.M. Yeung, and O.C. Au, "Fast Integer Motion Estimation for H.264 Video Coding Standard," *Proceeding of IEEE International Conference on Multimedia and Expo*, pp. I-289-292, 2004.

[17] H. Lee, H.J. Shim, Y. Park, and B. Jeon, "Fast Integer Motion Estimation for H.264 Video Coding Standard," *The Journal of The Korean Institute of Communication Sciences*, Vol. 39, No. 4, pp. 209-211, 2014.

[18] S.O. Kim, C.S. Park, H.J. Chun, and J.M. Kim, "A Fast and Low-complexity Motion Estimation for UHD HEVC," *Journal of Broadcast Engineering*, Vol. 18, No. 6, pp. 808-815, 2013.

[19] High Efficiency Video Coding Test Model 13.0, http://hevc.hhi.fraunhofer.de/ (accessed Oct., 3, 2017)

[20] N.S.K. McCann, W.J. Han, I.K. Kim, and J.H. Min, *Samsung's Response to the Call for Proposals on Video Compression Technology*, JCTVC-A124, 2010.

[21] G. Bjontegaard, *Calculation of Average PSNR Differences between RD Curves*, VCEG-M33, 2001.

이 규 중

2002년 서울대학교, 전자공학과 학사
2008년 University of Southern California, 전자공학과 석사
2013년 서울대학교, 전기컴퓨터공학부 박사
2013년~2017년 삼성전자 S.LSI 책임 연구원
2017년~현재 선문대학교 전자공학과 교수
관심분야 : 딥러닝, 영상 처리, 영상 압축, 멀티미디어 SOC 설계