특집논문 (Special Paper)
방송공학회논문지 제25권 제3호, 2020년 5월 (JBE Vol. 25, No. 3, May 2020)
https://doi.org/10.5909/JBE.2020.25.3.305
ISSN 2287-9137 (Online) ISSN 1226-7953 (Print)

# 적응적 움직임 벡터 해상도 고속 결정 기법

박 상 효[a)‡]

# Fast Decision Method of Adaptive Motion Vector Resolution

Sang-hyo Park[a)‡]

## 요 약

기존 동영상 부호화 표준보다 더 높은 효율의 표준에 대한 수요가 커지면서, 최근 MPEG과 VCEG에서 Versatile Video Coding(VVC)이라는 차세대 동영상 부호화 프로젝트를 개발하고 표준화하고 있다. 압축 효율 증대를 위하여 다양한 화면간 부호화 기법이 등장하였으며, 특히 움직임 벡터의 적응적인 해상도 부호화가 등장하여 VVC의 압축 효율을 올리는데 기여하였다. 다만, 최적의 움직임 벡터 해상도를 결정하기 위해 부호화기에서 다양한 율-왜곡 비용을 계산해야 했기에, 부호화기 시간 복잡도가 높아지게 되었다. 실시간 동영상 방송 및 스트리밍 서비스를 위해서는 부호화기의 복잡도를 줄이는 것이 필요하나, 아직 적응적 움직임 벡터 해상도 결정기법에 대한 복잡도 감소 연구는 미개척분야이다. 따라서, 본 논문에서는 이 움직임 벡터 해상도 결정을 위한 부호화 복잡도를 줄이는 연구를 제안한다. 이를 위해, VVC의 특별한 트리 구조인 multi-type tree 구조 내에서의 부호화된 문맥을 활용한 고속 결정기법을 고안한다. 실험 결과, 본 고속결정 기법은 VVC 참조 소프트웨어 대비 약간의 압축효율 감소 내에서 10%의 전체 부호화 시간을 줄임을 확인하였다.

## Abstract

As a demand for a new video coding standard having higher coding efficiency than the existing standards is growing, recently, MPEG and VCEG has been developing and standardizing the next-generation video coding project, named Versatile Video Coding (VVC). Many inter prediction techniques have been introduced to increase the coding efficiency, and among them, an adaptive motion vector resolution (AMVR) technique has contributed on increasing the efficiency of VVC. However, the best motion vector can only be determined by computing many rate-distortion costs, thereby increasing encoding complexity. It is necessary to reduce the complexity for real-time video broadcasting and streaming services, but it is yet an open research topic to reduce the complexity of AMVR. Therefore, in this paper, an efficient technique is proposed, which reduces the encoding complexity of AMVR. For that, the proposed method exploits a special VVC tree structure (i.e., multi-type tree structure) to accelerate the decision process of AMVR. Experiment results show that the proposed decision method reduces the encoding complexity of VVC test model by 10% with a negligible loss of coding efficiency.

Keyword : VVC, HEVC, inter prediction, motion vector resolution, encoder complexity

Copyright Ⓒ 2020 Korean Institute of Broadcast and Media Engineers. All rights reserved.
“This is an Open-Access article distributed under the terms of the Creative Commons BY-NC-ND (http://creativecommons.org/licenses/by-nc-nd/3.0) which permits unrestricted non-commercial use, distribution, and reproduction in any medium, provided the original work is properly cited and not altered.”

## Ⅰ. 서 론

High Efficiency Video Coding(HEVC)의 첫 표준안이 2013년에 나온 이후로도, 더 높은 화질의 영상에 대한 수요와 더불어, 실감미디어(예컨대, 360도 동영상)라는 새로운 미디어 포맷에 대한 수요가 증가하기 시작하였다. HEVC 표준화를 진행했을 당시에 비교하여, 주 영상 해상도는 훨씬 높아지게 되었고, 360도 동영상처럼 훨씬 많은 화소가 담긴 영상을 압축하기 위해서는 보다 획기적인 압축 표준이 필요하게 되었다. 이러한 수요를 바탕으로 MPEG과 VCEG 표준화단체들은 공동으로 Joint Video Exploration Team(JVET)을 결성하였으며, Versatile Video Coding(VVC)이라는 차세대 동영상 표준을 위한 초안을 2018년에 처음으로 발간하였다[1]. 그 이후에도, 지속적으로 진보된 기술들을 채택함으로써 VVC 참조소프트웨어(VVC test model, 이하 VTM)는 HEVC 참조소프트웨어(HEVC test model, 이하 HM) 대비 30% 이상의 압축 효율을 보인다고 보고된 바 있다[2].

VVC에 채택된 다양한 기술 중에는 동영상의 압축 효율을 획기적으로 끌어올리는 화면 간 예측(inter prediction)에 속하는 기술들도 다수 존재한다. 그 중, 새롭게 등장한 적응적 움직임 벡터 해상도(Adaptive motion vector resolution, 이하 AMVR) 기술은 동영상 콘텐츠의 특성에 따라 움직임 벡터 차이값 부호화/복호화를 위한 비트를 효과적으로 줄일 수 있게 하는 기법이다. HEVC에서는 휘도 성분을 위한 움직임 벡터(Motion vector, 이하 MV)를 부호화/복호화할 때, 정수단위 외에 1/2 및 1/4 화소 단위로 정밀한 움직임을 예측할 수 있게 MV를 위한 비트를 상세하게 전송할 수 있게 하였다. 그러나 최대 1/4 화소까지 보내려다 보니, 움직임이 큰 동영상의 경우, 매우 큰 MV값이 나타날 수 있으며, 이에 따라 MV를 위한 비트가 증가할 가능성이 높아져, 압축 효율의 저하를 이끌 수 있다. 이러한 동영상 특성에 따라 보다 효율적으로 MV를 압축하기 위하여, 때로는 HEVC처럼 1/4 화소 단위로 혹은 정수단위로 MV를 적응적으로 보낼 수 있게 하는 기법이 바로 AMVR이다[3].

이렇게 AMVR을 통하여 동영상 특성에 맞게 비트율을 효과적으로 줄이기 위해서는, 어떠한 MV 해상도로 MV를 부호화해야 할지 정해야 하기 때문에, 부호화기는 매번 다양한 해상도의 MV에 대한 테스트를 진행하는 것이 필요하다. 즉, 매 블록을 예측할 때 AMVR을 위한 율-왜곡 비용(rate-distortion cost)을 계산해야 하므로, 부호화기의 시간 복잡도가 증가하는 것은 필연적이다. 그러나, 실시간으로 동영상을 촬영하여 방송하거나 스트리밍 서비스를 제공해야 하는 기기, 또는 실시간으로 녹화/전송을 해야 하는 드론이나 감시카메라에게 이러한 부호화 복잡도는 큰 걸림돌이 될 수 있다. JVET에서도 이러한 부호화 복잡도를 해결하기 위하여 여러 노력을 하였으나, 아직까지 AMVR의 복잡도를 개선하기 위한 획기적인 연구는 미미한 실정이다.

특별히, VVC의 이전 모델들이라 할 수 있는 HM과 Joint Exploration Test Model (JEM)에서의 움직임 예측 관련 부호화 성능 향상을 위한 연구가 있다[12][13][14]. 그러나 VVC에 대한 AMVR관련 획기적인 연구는 아직 미미하며, 다만 한 Ad-Hoc Group에서 회의마다 꾸준히 AMVR에 대한 계산 복잡도와 압축 성능에 관하여 평가 결과를 발표를 한 바 있다[4].

본 논문에서는 AMVR의 부호화 복잡도를 효과적으로 감소하기 위한 고속 결정 기법을 제안한다. 이를 위하여, VVC의 대표적 특징으로 꼽는 multi-type tree(MTT) 구조를 활용하는 문맥 기반 고속 결정 기법을 제안한다. 특히 기 부호화된 움직임 및 부호화 유닛 내 문맥 정보를 활용함으로써, AMVR이 불필요한 블록을 조기에 판단하여 부호화기 복잡도를 낮춘다. JVET에서 권고하는 공통 시험 조건(common test condition, 이하 CTC)에 따른 다양한 영상들에 대하여 실험을 진행하여, 본 제안기법의 압축 성능 대비 부호화 복잡도 감소율에 대한 효율성을 검증하였다. 그 결과, 본 제안기법은 VTM 6.0 대비 0.48%의 압축효율 감소 가운데, 부호화 복잡도를 평균 90%로 감소시킴을 확인할 수 있다.

a) 경북대학교 컴퓨터학부(School of Computer Science and Engineering, Kyungpook National University)
‡ Corresponding Author : 박상효(Sang-hyo Park)
E-mail: s.park@knu.ac.kr
Tel: +82-53-950-6373
ORCID: https://orcid.org/0000-0002-7282-7686
· Manuscript received March 23, 2020; Revised May 11, 2020; Accepted May 11, 2020.

본문은 다음과 같이 구성한다. 2장에서는 VVC의 주요 특징이라 할 수 있는 MTT 구조에 대한 개요와 AMVR에 대한 관련 연구를 분석하고 정리한다. 3장에서는 MTT 구조에 대한 제안기법과 이를 뒷받침하는 통계자료를 제시한다. 4장에서는 제안기법의 성능을 평가하고 이에 대한 관찰 결과를 보고한다. 마지막으로 5장에서는 본문을 결론짓는다.

## Ⅱ. 관련 연구

HEVC에서는 부호화 유닛(coding unit, 이하 CU)과 예측 유닛(prediction unit, 이하 PU)의 크기와 모양이 구분될 수 있었다[9]. 예컨대, CU가 32 × 32의 크기일 경우, PU의 크기는 CU와 동일한 모양과 크기이거나(32 × 32) 1 : 3, 2 : 2, 3 : 1의 크기로 가로 방향 및 세로 방향으로 나누어질 수 있었다. 그러나 VVC에서는 CU와 PU의 개념을 합치는 대신, CU의 분할 모양과 크기를 훨씬 다양하게 할 수 있는 MTT구조를 취한다. 여기서 MTT는 HEVC에서 사용한 4진 트리 외에 2진 트리와 3진 트리를 추가한 형태이다. 단, HEVC처럼 Root는 4진 트리부터 시작하는 것을 기본 골자로 하며, 4진 트리의 마지막 노드일 경우(더 이상 4진 트리로 나누지 않는 명시적/암시적 경우)에만 2진 트리 혹은 3진 트리의 구조를 취할 수 있다. 이렇게 4진 트리가 아닌 구조가 시작할 경우, 2진 트리의 자식 노드는 2진 트리일 수도 있고 3진 트리일 수도 있다. 그리고 2진/3진 트리 모두 가로 방향으로 분할할 지 혹은 세로 방향으로 나누는 경우에 대한 명시적으로 플래그(flag)를 전달함으로써 CU의 최종 모양과 크기를 알 수 있게 된다.

VVC는 이렇게 CU 모양/크기의 다변화를 꾀함으로써 HEVC보다 높은 압축 성능을 갖게 되었다. 또한, AMVR이라는 움직임 벡터 차(motion vector difference, 이하 MVD)에 대한 부호화 과정 역시 보다 효율적으로 압축할 수 있는 새로운 기법이 VVC에 추가되었다. HEVC에서는 슬라이스 헤더에서 정수 움직임 벡터(motion vector, 이하 MV)를 사용하라는 명시적 플래그가 없다면, 1/4화소 단위로 MVD값을 부호화한다. 반면, VVC에서는 CU단위에서 AMVR 사용을 가능하게 하여, MVD 값의 단위를 부호화기가 보다 다양하게 선택할 수 있게 하며, 이에 따라 압축 성능 향상을 이끈다.

AMVR 모드는 크게 2가지로 나뉜다. 일반적인 AMVR 모드와 아핀(affine) AMVR 모드가 있다. 먼저 일반적인 AMVR 모드에서는 부호화기는 총 4가지의 MVD 해상도를 적응적으로 선택하여 부호화할 수 있다. 1/4 휘도 화소, 1/2 휘도 화소, 정수단위 휘도 화소, 4배수 정수단위 휘도 화소이다. 여기서 4배수 정수단위 휘도 화소라 함은 부호화된 MVD값을 복호화할 경우, 실제로는 4배의 값으로 복호화해야 함을 의미한다. 이는 객체와 카메라의 움직임이 매우 커서 MV 값이 커질 수밖에 없는 영상에 유효하다. 아핀 AMVR 모드는 일반적인 AMVR 모드와 약간 다르다. 1/4 휘도 화소, 정수단위 휘도 화소, 1/16 휘도 화소의 총 3가지 모드가 존재한다. 아무래도 아핀으로 MV를 부호화한다는 경우는 보다 정밀한 움직임이 필요한 경우일 수도 있기에 일반적인 AMVR 보다는 더 촘촘한 단위까지 MVD를 전송할 수 있는 것이 유리할 수 있다.

어떤 MV 해상도로 부호화를 하는 것이 해당 CU를 압축하는 효율성 측면에서 유리한지 알기 위해서는, 모든 경우에 대해서 율-왜곡 비용을 계산해야 한다. 그러나 이는 부호화 시간 복잡도 측면에서 너무 비효율이기 때문에, VTM 6.0의 경우 1/4 휘도 화소 외의 나머지 해상도 단위에 대해서는 조건부 비용 계산 알고리즘이 포함되어 있다[3]. 예컨대, 한 CU가 적어도 하나의 non-zero MVD 값을 갖고 있다면, 그 CU의 1/4 휘도 화소 MVD 해상도를 먼저 갖는 것으로 한다. 이에 해당하지 않을 경우에 다른 MVD 해상도인 1/2 휘도 화소, 정수 단위 휘도 화소, 4배수 정수단위 휘도 화소에 대해 추가로 결정 여부를 정한다. 특히, 일반적인 AMVR 모드의 경우, 가장 먼저는 1/4 휘도 화소의 율-왜곡 비용과 정수단위 회도 화소의 율-왜곡 비용을 비교하여, 그 비용 비교에 따라 4배수 정수단위 회도 관련 비용 계산을 할지 말지를 결정한다. 여기서 율-왜곡 비용은 왜곡 비용 함수인 Sum of Absolute Difference(SAD) 혹은 Hadarmard transformed SAD(SATD)를 통해 계산하며 이에 대한 식은 다음 (1)과 (2)와 같다.

$$SAD = \sum_{i,j} |d(i,j)| \tag{1}$$

$$SATD = (\sum_{i,j} |dT(i,j)|) / 2 \tag{2}$$

여기서 i와 j는 같은 크기의 비교 대상인 두 블록의 x축과 y축 좌표를 의미하며, d(i, j)는 두 블록의 차이값을 의미하고, dT(i, j)는 두 블록의 차이값을 추측하기 위하여 Hadamard 변환으로 추측하는 차이값을 의미한다. 이를 통해 율-왜곡 비용 J를 아래와 같이 구할 수 있다. SAD로 계산한 JSAD와 SATD로 계산한 JSATD는 각각 (3)과 (4)와 같다.

$$J_{SAD} = SAD + \lambda R \tag{3}$$

$$J_{SATD} = SATD + \lambda R \tag{4}$$

여기서 R은 모드 결정을 위해 필요로 하는 비트 비용을 의미하여, 상수 λ는 서로 다른 단위인 R과 왜곡 비용을 Lagrangian 기반의 최적화를 위하여 고안된 lambda 값이며, 이 값은 양자화 파라미터(QP)에 지수적으로 비례하여 결정된다. 만약 정수단위 관련 비용이 1/4 휘도 관련 비용보다 훨씬 크다면 4배수 단위 역시 비용이 클 것이라 가정하여 해당 비용 관련 계산 과정을 생략하는 것이다. 이와 비슷하게, 만약 정수단위 관련 비용이 이전에 테스트된 단위 관련 비용보다 매우 클 경우, 1/2 휘도 관련 비용 계산 과정을 생략한다.

이러한 율-왜곡 비용 기반 생략기법을 적용하여도, 여전히 VTM 내에서의 AMVR의 부호화 복잡도는 상당하다. 전체 부호화기 복잡도 대비 임의접근(Random-access) 시나리오에서 19%, 저지연(Low-delay) 시나리오에서 12%나 차지한다고 보고된 바 있다[4]. 약 24개 이상의 부호화 툴(Tool)이 적용되고 있는 VTM 내에서도 하나의 툴이 10% 이상의 부호화 복잡도를 차지하는 경우는 희소한 경우이며, 그만큼 AMVR이 VTM의 부호화 복잡도를 높이는 주요 원인으로 볼 수 있다.

VVC의 부호화 복잡도, 특히 화면 간 예측의 복잡도를 줄이기 위해, 최근 다양한 연구자들이 관심을 갖고 연구하는 열린 분야이다. 특히, 기존의 HEVC에 없던 툴에 대하여 연구가 집중되고 있다. 그 중의 하나인 아핀 움직임 예측에 대하여 부호화 중에 얻을 수 있는 MTT 기반 정보를 활용하여 고속으로 생략하는 기법이 있다[5]. 또한 Symmetric MVD(SMVD)의 부호화기 복잡도 개선을 위하여 초기 MV 후보군의 최대 개수를 5로 제한하고, SMVD 왜곡 계산과정을 고속화한 기법이 JVET에서 발표되었다[6]. 그 외에도, 휘도와 색차 신호의 움직임 보상(motion compensation) 과정을 분리함으로써, subblock merge 후보군을 위한 색차 신호 왜곡 비용 계산 과정을 생략하는 기법이 JVET에서 발표되었다[7]. SIMD 병렬화 기법을 통해 움직임 보상 과정에 대한 부호화 복잡도를 줄이는 기법 또한 JVET에서 발표되었다[8].

HEVC에서는 부모 노드 CU의 정보가 자녀 노드 CU의 모드 결정에 결정적인 정보를 제공할 수 있는 경우가 있음이 밝혀졌다[10][11]. 특히 움직임 예측 성능을 개선하기 위한 정보로서도 효과적임이 HEVC의 4진 트리에 더불어 JEM의 4진 및 2진 트리에서도 연구된 바 있다. 다만, 이러한 기존 연구들은 모두 MTT에 대한 검증이 없으며, 일반적인 모드 결정 방법 혹은 전통적인 움직임 예측 관련 MV 결정 방법만을 다루고 있다.

그러나 최신 VTM을 위한 AMVR의 고속화 기법이나 최적화 기법에 대한 연구는 거의 발표된 바가 없다. 따라서, 본문에서 AMVR을 고속화하는 기법을 제안함으로써 향후 실시간 혹은 저복잡도의 제한이 있는 환경에서의 VVC 부호화기 어플리케이션을 구현함에 기여하고자 한다.

## Ⅲ. 제안 기법

본문에서는 AMVR을 고속으로 결정하기 위하여, MTT 기반 기 부호화된 정보 활용 방법을 제안한다. MTT의 정보와 AMVR 모드에 대한 상관관계를 유추하기 위하여, 본문에서는 부모 노드 CU에서의 최선의 AMVR 모드와 MTT 트리 구조 등에 대한 정보가 미리 주어질 때, 자녀 노드 CU에서 AMVR을 결정할 때 통계적으로 얼마나 도움이 되는지 알기 위하여 여러 부호화 문맥에 대한 확률을 조사한다. 이러한 부모 노드와 자녀 노드 간의 정보 전달 과정을 MTT 구조에 맞게 예시로 그리면 그림 1과 같다. MTT에는 총 3가지가 있으며 2진 트리(binary tree)와 3진 트리(ternary tree)의 경우, 가로/세로 방향의 분할방향을 정해주어야 한다. 본 그림 1은 2진 트리의 세로 방향에 대한 자녀 노드를 예시로 표현하였으며, 여기에서 BT_V는 2진 트리 세로 방

향 분할, BT_H는 2진 트리 가로 방향 분할, TT_V는 3진 트리 세로 방향 분할, TT_V는 3진 트리 가로 방향 분할을 의미한다.

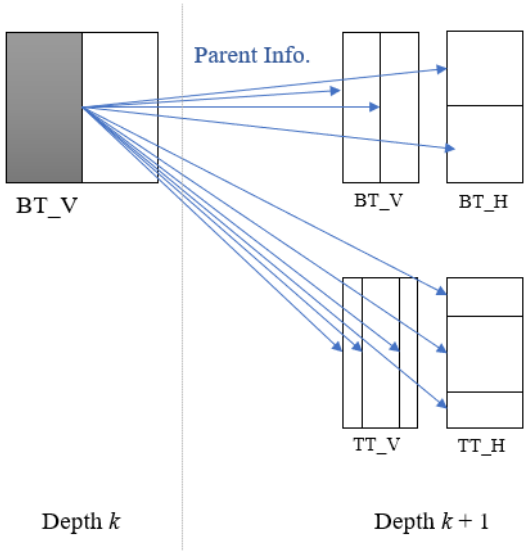


그림 1. MTT에서의 부모 노드와 자녀 노드 간 정보 전달 관계 예시
Fig. 1. Example of information relationship delivered between parent and child nodes in an MTT structure

AMVR과 부호화 중의 MTT 정보를 확인하기 위해서는 VTM의 부호화 과정에서 정보를 얻어야 한다. 이를 위하여, 다음의 환경과 조건 하에 VTM 부호화 실험을 진행하였다. 먼저 VTM 6.0에서 통계를 위한 코드를 추가하였으며, 부호화 환경설정은 저지연 시나리오, 양자화 파라미터(Quantization parameter, 이하 QP) 30, 총 20프레임 부호화로 설정하였다. 본 통계값 추출을 위해 사용한 테스트 영상은 JVET 공통시험조건(common test condition, 이하 CTC)[15]에서 규격하는 영상에서 되도록 다양한 해상도를 반영하여 선정하였으며, 다음과 같다 :

MarketPlace, Cactus, BQTerrace, BQMall,
RaceHorsesC, KristenAndSara.

확률값을 정의하기 위해 본문에서는 다음 용어들을 먼저 정의한다. 부모 노드의 최적의AMVR 모드를 $A_{par}$, 자녀 노드(현재 노드)의 최적의 AMVR 모드를 $A_{cur}$라고 정의하겠다. $A_{par}$가 존재할 때를 $\forall A_{par}$라고 정의하며, $A_{cur}$가 존재할 때를 $\forall A_{cur}$라고 정의하겠다. 여기서 AMVR 모드가 존재한다는 의미는, 해당 CU가 AMVR 모드를 테스트하기 위하여 율-왜곡 비용을 계산한 경우를 의미한다. 또한 여기서 AMVR 모드라 함은 AMVR의 모든 모드(1/4 휘도, 1/2 휘도 모드 등)를 의미한다. 만약 현재 노드(자녀 노드)에서 부모 노드의 AMVR 모드 정보를 미리 알 수 있는 경우에 $A_{par}$와 $A_{cur}$가 같은 경우의 수를 $n(A_{par} = A_{cur} \mid \forall A_{par})$라고 할 수 있고, 이에 대한 조건부 확률 $p(A_{par} = A_{cur} \mid \forall A_{par})$를 구할 수 있다. 또한 부호화 중에는 부모 노드의 최적의 AMVR 모드가 존재하지 않는 경우, 즉 $\forall(\sim A_{par})$, 그리고 이와 비슷하게 자녀 노드의 최적의 AMVR 모드가 존재하지 않는 경우, 즉 $\forall(\sim A_{cur})$가 동시에 존재하는 경우, 즉 $p(\forall(\sim A_{par}) \cap \forall(\sim A_{cur}))$가 존재할 수 있다. 이러한 부모 노드/자녀 노드의 AMVR 정보 뿐 아니라, 부모 노드와 자녀 노드가 각각 속한 MTT 구조는 무엇인지 역시 조사할 수 있다. 이때, 부모 노드의 트리 구조 및 분할 방향(예컨대 BT_V, TT_H 등)을 $T_{par}$라고 정의하며, 자녀 노드의 트리 구조 및 분할 방향을 $T_{cur}$라고 정의하겠다. 그러할 경우, 이러한 트리 구조 및 분할방향과 AMVR 최적 모드와의 확률을 조사할 수 있다. 먼저, $T_{par}$와 $T_{cur}$가 같은 상황에서 $A_{par}$와 $A_{cur}$가 같은 경우의 확률, 즉 $p(T_{par} = T_{cur} \cap A_{par} = A_{cur})$를 구할 수 있다.

부모 노드와 자녀 노드 간의 확률값들과 샘플 수에 대하여 표 1에서와 같이 조사하였다. 여기서 샘플이라 함은, 하기 서술할 각 조건에 부합하는 CU의 개수를 의미하며, 이 CU의 개수를 세기 위하여 VTM의 기본(default) 부호화 환경에서 부호화를 진행할 때 율-왜곡 비용이 계산된 모든 CU를 의미한다. 표 1에서 보면 확률값이 높은 경우는 $p(A_{par} = A_{cur} \mid \forall A_{par})$와 $p(T_{par} = T_{cur} \cap A_{par} = A_{cur})$이다. 전자는 0.788, 후자는 0.796의 확률값을 평균적으로 갖는다. 다만 후자의 경우, 샘플 수 자체가 적어서 이를 활용할 수 있는 경우가 제한적이다(전자 대비 약 27% 수준). 따라서 본문에서는 샘플 수가 많아서 활용 가능성이 높으면서 확률값이 높은 $p(A_{par} = A_{cur} \mid \forall A_{par})$에 기반하여, 자녀 노드에서의 AMVR과정을 획기적으로 생략하는 방법을 제안한다.

표 1. 부모 노드와 자녀 노드간의 AMVR 최적의 모드, 트리 구조, 분할 방향에 따른 통계
Table 1. Statistics of optimal AMVR mode, tree structure, and split direction corresponding to the relationship between parent and child nodes

| Sequence | $n(A_{par} = A_{cur} \mid \forall A_{par})$ | $n(\forall(\sim A_{par}) \cap \forall(\sim A_{cur}))$ | $n(T_{par} = T_{cur} \cap A_{par} = A_{cur})$ | $p(A_{par} = A_{cur} \mid \forall A_{par})$ | $p(\forall(\sim A_{par}) \cap \forall(\sim A_{cur}))$ | $p(T_{par} = T_{cur} \cap A_{par} = A_{cur})$ |
|---|---|---|---|---|---|---|
| MarketPlace | 2,507,074 | 370,242 | 656,372 | 0.803 | 0.608 | 0.793 |
| Cactus | 3,198,376 | 469,127 | 837,811 | 0.807 | 0.575 | 0.807 |
| BQTerrace | 2,603,098 | 358,813 | 721,089 | 0.813 | 0.525 | 0.83 |
| BQMall | 756,108 | 100,610 | 204,649 | 0.774 | 0.532 | 0.791 |
| RaceHorsesC | 1,502,153 | 215,110 | 407,880 | 0.657 | 0.47 | 0.697 |
| KristenAndSara | 398,158 | 52,966 | 105,123 | 0.875 | 0.642 | 0.855 |
| Average | 1,827,495 | 261,145 | 488,821 | 0.788 | 0.559 | 0.796 |

낸 경우), 본본문에서 제안하는 기법은 MTT 구조에서의 부호화가 선행된 부모 노드의 데이터를 활용하여 고속으로 AVMR의 최적의 모드를 결정하는 기법을 제안한다. 특별히 부모 노드의 최적의 AMVR 모드가 존재하는 경우(즉, 부모 노드 CU가 AMVR을 테스트하여 최적의 값을 찾아문의 제안기법은 부모 노드의 자녀 노드인 현재 CU에서의 AMVR 과정을 생략 할 수 있다. 제안기법을 도식화하면 그림 2와 같다. 먼저 부모 노드의 최적의 AMVR 정보가 있을 경우와 없을 경우로 나뉜다. 만약 MTT 구조에서의 부모 노드의 최적의 AMVR 정보, 즉 $A_{par}$가 없을 경우에는 일반적인 AMVR 결정 과정을 진행한다. 여기서, 초기의 AMVR값을 Acur_init라고 하겠다. $A_{cur_init}$은 최적의 값이 나올 때까지 임시로 수행하는 남아있는 AMVR 모드들 중 하나를 의미한다. 이를 통해 현재 CU의 최적의 AMVR 정보, 즉 $A_{cur}$를 추후 저장하여, 본 CU의 자녀 노드 CU에게 그림 1에서와 같이 $A_{cur}$ 정보를 전달한다. 이 $A_{cur}$는 자녀 노드 CU 입장에서는 부모로부터의 정보(즉, $A_{par}$)가 된다. 만약 부모 노드의 최적의 AMVR 정보, 즉 $A_{par}$가 있을 경우에는, $A_{cur_init}$가 $A_{par}$와 같은지를 비교한다. 만약 같은 값이면, 현 CU의 AMVR 과정은 본래 모든 가능한 모드에 대해 모두 테스트를 해야 하나, $A_{cur_init}$에 대한 테스트만 진행하고, $A_{cur_init}$ 모드를 최적의 $A_{cur}$로 가정하여 부호화를 진행한

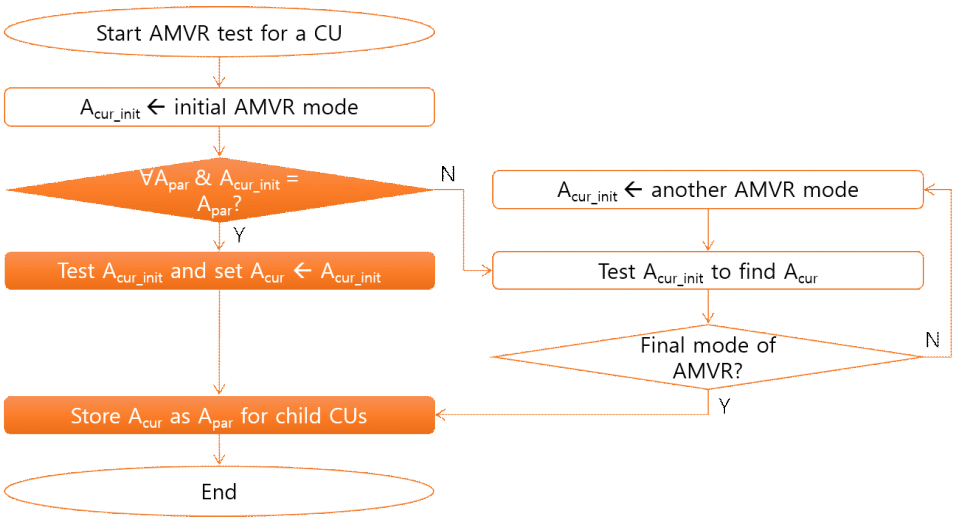


그림 2. CU 부호화 과정 중에서의 제안기법(기존 VTM에 없는 부분을 음영처리함)
Fig. 2. The proposed method during a CU encoding (The area in orange indicates the proposed method)

다. 즉, $A_{cur_init}$가 $A_{par}$와 같은지를 비교함으로써, $A_{cur_init}$가 아닌 나머지 AMVR 모드들에 대한 테스트 과정(율-왜곡 비용 계산 등)을 생략하여 부호화 복잡도를 효과적으로 줄일 수 있다.

## Ⅳ. 실험 결과

제안기법의 우수성을 평가하기 위하여, VTM 6.0 소프트웨어와의 성능 비교를 실험하였다. 본 실험은 CTC 설정에 맞추어, 저지연 시나리오와 이에 해당하는 필수 테스트 영상 20개 중에, 표 1에서 통계를 위해 사용한 6개의 영상을 제외하였으며, 또한 스크린 콘텐츠를 위한 영상 4개 역시 제외하였다. 사용한 프레임은 각 영상별 초당 프레임(frame per second) * 2에 해당하는 숫자 만큼 부호화를 진행하였다. 또한, 압축성능의 비교를 위하여, 동일한 화질에서의 비트율 차이값을 계산하는 BD-rate[16]으로 휘도성분을 평가하였다. 부호화 복잡도는 실행시간(encoding time, 이하 ET)으로 측정하였다. ET는 수식 (5)와 같다.

$$ET = \frac{T_{proposed}}{T_{anchor}} \tag{5}$$

여기서 $T_{anchor}$는 VTM 6.0의 전체 부호화 시간을, $T_{proposed}$는 제안기법의 전체 부호화시간을 의미한다. 실험 환경은 64비트 Windows 10이 탑재된 PC이며, PC의 주요 사양은 Intel i3-8100 (3.60GHz 4코어) 및 8GB RAM이다.

실험 결과는 표 2와 같다. 제안기법은 VTM 6.0 대비 평균, 0.48%의 BD-rate 증가와 더불어 ET를 90%로 감소시켰다. 가장 압축률 손실이 큰 경우는 BasketballDrive로, 움직임이 무척 크고 카메라 움직임도 큰 영상이다. 대신 이러한 경우, ET의 시간 절감 폭 역시 큰 것으로 나타났다. 흥미로운 점은, BQSuare와 같이, 오히려 제안기법을 통해 압축률이 개선되는 영상도 존재한다는 점이다. 압축률을 개선하면서 동시에 ET 역시 7%나 줄이는 것을 확인하였다. 이처럼 제안기법은 기존 VTM의 압축률을 합리적으로 거의 유지하면서, 부호화 시간을 유의미하게 줄이는 것을 확인하였다.

표 2. VTM과 제안기법의 압축률 및 부호화 시간 성능비교
Table 2. Performance comparison of VTM and the proposed method in view of coding efficiency and encoding time

| Resolution | Sequence | BD-rate | ET |
|---|---|---|---|
| 1920 × 1080 | RitualDance | 0.72% | 89% |
| 1920 × 1080 | BasketballDrive | 0.97% | 88% |
| 832 × 480 | BasketballDrill | 0.79% | 89% |
| 832 × 480 | PartyScene | 0.32% | 90% |
| 416 × 240 | BasketballPass | 0.58% | 89% |
| 416 × 240 | BQSquare | -0.12% | 93% |
| 416 × 240 | BlowingBubbles | 0.38% | 89% |
| 416 × 240 | RaceHorses | 0.42% | 88% |
| 1280 × 720 | FourPeople | 0.31% | 91% |
| 1280 × 720 | Johnny | 0.41% | 92% |
| **Average** | | **0.48%** | **90%** |

## Ⅴ. 결 론

본 논문에서는 차세대 동영상 부호화 표준으로 주목받는 VVC의 핵심 툴 중의 하나인 AMVR에 대하여 다루었다. 특히, AMVR의 부호화 복잡도 문제를 유의미하게 개선하는 제안기법을 제안하였으며, 이는 VVC의 특징 중 하나인 MTT 구조를 활용하였다는 점에서 기존 연구들과의 차별성이 있다. 또한, 제안기법을 적용하였을 때 영상에 따라서는 압축률이 오히려 개선되는 부분도 나타났다. 이에 따라, AMVR에 대한 압축 성능과 부호화 복잡도를 모두 개선할 수 있는 부분이 있다고 본다. 또한, 제안기법은 실시간 영상 스트리밍/방송이 필요한 영상통화와 같은 저지연 시나리오가 필요한 부호화 어플리케이션에 적용할 때, VVC 부호화 복잡도를 효과적으로 줄일 수 있으리라 믿는다. 향후, AMVR의 시간/공간 복잡도와 VVC 내의 복잡도 상관관계를 분석하여 다른 툴과의 영향력과 추가 개선할 여지를 찾는 연구를 진행할 계획이다.

## 참 고 문 헌 (References)

[1] B. Bross, “Versatile Video Coding (Draft 1),” Joint Video Experts Team (JVET) of ITU-T SG 16 WP 3 and ISO/IEC JTC 1/SC 29/WG 11, Document JVET-J1001, Apr. 2018. http://phenix.it-sudparis.eu/

jvet/ (accessed May 6, 2020)

[2] F. Bossen, X. Li, K. Suehring, and A. Norkin, “AHG report: Test model software development (AHG3),” Joint Video Experts Team (JVET) of ITU-T and ISO/IEC, Document JVET-N0003, Mar. 2019. http://phenix.it-sudparis.eu/jvet/ (accessed May 6, 2020)

[3] J. Chen, Y. Ye, and S. H. Kim, “Algorithm description for Versatile Video Coding and Test Model 6 (VTM 6),” Joint Video Experts Team (JVET) of ITU-T and ISO/IEC, Document JVET-O2002-v2, Jul. 2019. http://phenix.it-sudparis.eu/jvet/ (accessed May 6, 2020)

[4] W.-J. Chien, J. Boyce, W. Chen, Y.-W. Chen, R. Chernyak, K. Choi, R. Hashimoto, Y.-W. Huang, H. Jang, R.-L. Liao, S. Liu, “JVET AHG report: Tool reporting procedure (AHG13),” Joint Video Experts Team (JVET) of ITU-T and ISO/IEC, Document JVET-OP0013, Oct. 2019. http://phenix.it-sudparis.eu/jvet/ (accessed May 6, 2020)

[5] S. Park and J. Kang, "Fast Affine Motion Estimation for Versatile Video Coding (VVC) Encoding," IEEE Access, vol. 7, pp. 158075-158084, 2019. 10.1109/ACCESS.2019.2950388

[6] H. Chen, H. Yang, “AHG13: Encoder speed-up for SMVD,” Joint Video Experts Team (JVET) of ITU-T and ISO/IEC, Document JVET-P0092, Oct. 2019. http://phenix.it-sudparis.eu/jvet/ (accessed May 6, 2020)

[7] G. Li, X. Li, X. Xu, S. Liu, "Non-CE4: Encoder optimization for sub-block-based merge candidate search," Joint Video Experts Team (JVET) of ITU-T and ISO/IEC, Document JVET-P0445, Oct. 2019. http://phenix.it-sudparis.eu/jvet/ (accessed May 6, 2020)

[8] X. Xiu, Y.-W. Chen, T.-C. Ma, H.-J. Jhu, X. Wang, "Non-CE4: SIMD support for motion compensated prediction at high internal bit-depth,” Joint Video Experts Team (JVET) of ITU-T and ISO/IEC, Document JVET-P0512, Oct. 2019. http://phenix.it-sudparis.eu/jvet/ (accessed May 6, 2020)

[9] G. J. Sullivan, J. Ohm, W. 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, Dec. 2012. 10.1109/TCSVT.2012.2221191

[10] K. Choi, S. -H. Park, E. S. Jang, “Coding tree pruning based CU early termination,” Joint Collaborative Team on Video Coding (JCT-VC) of ITU-T and ISO/IEC, Document JCTVC-F092, Jul. 2011. http://http://phenix.int-evry.fr/jct/ (accessed May 6, 2020)

[11] S. Park, S. Lee, E. S. Jang, D. Jun, and J. -W. Kang "Efficient biprediction decision scheme for fast high efficiency video coding encoding," Journal of Electronic Imaging, vol. 25, no. 6, 063007 Nov. 2016. 10.1117/1.JEI.25.6.063007

[12] Z. Pan, J. Lei, Y. Zhang, X. Sun and S. Kwong, "Fast Motion Estimation Based on Content Property for Low-Complexity H.265/HEVC Encoder," IEEE Transactions on Broadcasting, vol. 62, no. 3, pp. 675-684, Sept. 2016. 10.1109/TBC.2016.2580920

[13] S. Park and E. S. Jang, "Comments on “Fast Motion Estimation Based on Content Property for Low-Complexity H.265/HEVC Encoder”," IEEE Transactions on Broadcasting, vol. 63, no. 4, pp. 740-742, Dec. 2017. 10.1109/TBC.2017.2711146

[14] S. Park and E. S. Jang, "An Efficient Motion Estimation Method for QTBT Structure in JVET Future Video Coding," 2017 Data Compression Conference (DCC), Snowbird, UT, 2017, pp. 454-454. 10.1109/DCC.2017.62

[15] F. Bossen, J. Boyce, K. Suehring, X. Li, and V. Seregin, “JVET common test conditions and software reference configurations for SDR video,” Joint Video Experts Team (JVET) of ITU-T and ISO/IEC, Document JVET-N1010, Mar. 2019. http://phenix.it-sudparis.eu/jvet/ (accessed May 6, 2020)

[16] G. Bjontegaard, “Calculation of average PSNR differences between RD-curves,” ITU-T SG16/Q, Document VCEG-M33, Mar. 2001. https://www.itu.int/wftp3/av-arch/video-site/ (accessed May 6, 2020)

## 저 자 소 개

**박 상 효**

- 2011년 2월 : 한양대학교 컴퓨터전공 (학사)
- 2017년 8월 : 한양대학교 컴퓨터·소프트웨어학과 (박사)
- 2017년 5월 ~ 2018년 2월 : 전자부품연구원 지능형영상처리센터 Post-doc.
- 2018년 3월 ~ 2018년 12월 : 연세대학교 바른ICT연구소 연구원
- 2019년 2월 ~ 2020년 1월 : 이화여자대학교 전자전기공학과 박사후연구원
- 2020년 3월 ~ 현재 : 경북대학교 컴퓨터학부 조교수
- ORCID : http://orcid.org/0000-0002-7282-7686
- 주관심분야 : HEVC, VVC, Encoding/Decoding Complexity, Omnidirectional Video, Deep Learning