

HEVC에서 차분 움직임 벡터 코딩을 위한 지수 골롬 코드의 적응적인 차수 선택 방법

*최광현 **한종기

세종대학교

*choi850603@naver.com **hjk@sejong.edu

Motion Vector Coding Using Adaptive Exp-Golomb Code

*Choi, Kwang-Hyun **Han, Jong-Ki

Sejong University

요약

HEVC 비디오 코덱에서는 인터 예측을 수행할 때 고정된 지수 골롬 코드를 사용하여 차분 움직임 벡터를 부호화한다. 그러나 고정된 Exp-Golomb 코드를 사용하게 되면 영상의 움직임을 고려하지 않고 영상의 국지적인 특성을 반영하는데 어려움이 있다. 본 논문에서는 현재 블록의 MVP와 예측된 MV에 따라 적응적인 Exp-Golomb 코드를 결정하는 방법을 제안한다. 현재 블록의 MVP에 따라 현재 MV를 예측하는 MV모델링을 통하여 Exp-Golomb 코드를 결정한다. HEVC의 참조소프트웨어인 HM6.0을 이용하여 실험한 결과, Random Access 부호화 구조에서는 평균적으로 약 0.9%의 성능 향상을 얻을 수 있었으며, Low Delay 부호화 구조에서는 약 0.6%의 BD-rate의 감소를 확인할 수 있었다.

1. 서론

국제 비디오 표준화 기구인 ISO/IEC SC29/WG11 MPEG(Moving Picture Expert Group)과 ITU-T VCEG (Video Coding Experts Group)이 공동으로 JCT-VC (Joint Collaborative Team on Video Coding)를 설립하였다. 그리고 새로운 비디오 코덱 표준화 프로젝트인 HEVC (High Efficiency Video Coding)^[1] 표준화를 진행 중에 있다.

MPEG-2, H.264 등의 기존의 코덱들은 16x16 MB (Macro block)으로 분할하여 부호화하였으나 HEVC에서는 부호화 기본단위를 CU(Coding Unit)를 사용한다. CU의 크기는 최대 64x64에서 최소 8x8 까지 분할 될 수 있다. 하나의 CU는 또다시 여러 개의 PU로 분할되어 예측된다. 예측 방법은 Intra와 Inter 예측으로 나눌 수 있다. Intra 예측은 최대 34가지 방향성을 고려하는 모드로 제안을 하고 있다. Inter 예측 역시 부호화 효율을 높이기 위해 AMVP, Merge 모드 등 여러 가지 기술을 사용하고 있다. Inter 예측 시 현재 블록과 가장 비슷한 블록을 찾는 과정을 움직임 추정이라고 한다. 움직임 추정 과정을 통하여 움직임 벡터를 찾게 된다. 이렇게 찾아진 움직임 벡터는 주변 블록의 움직임 벡터와의 차분을 부호화 하게 된다. HEVC에서 차분 움직임 벡터의 경우 1차 Exp-Golomb 코드를 사용하여 부호화하고 있다. 1차 Exp-Golomb 코드로 차분 움직임 벡터를 부호화하기 때문에 멀리 있는 움직임을 부호화할 때 상대적으로 높은 차수에 비해 불리하게 적용된다. 큰 차분 움직임 벡터를 부호화 하고 움직임이 많은 영상에서 좀

더 효과적으로 움직임을 예측하고 비디오 코덱의 부호화 효율을 향상시키는 기술이 필요하다.

본 논문에서는 최적의 지수 골롬 코드를 결정하기 위하여 주변의 MVP를 이용하여 MV를 예측하고, 예측된 MV로 MVD를 구하게 된다. 예측된 MVD로 지수 골롬 코드의 코드워드 길이가 가장 작은 지수 골롬 코드의 차수를 할당하여 부호화 한다.

본 논문의 구성은 다음과 같다. 제 2장에서는 기존 코덱의 움직임 벡터 부호화 방법을 소개한다. 제 3장에서는 제안하는 차분 움직임 벡터 코딩을 위한 지수 골롬 코드의 적응적인 차수 선택 방법을 서술한다. 제 4장에서는 제안하는 알고리즘을 적용한 실험결과 및 분석을 서술하고, 제 5장에서 본 논문의 결론을 맺는다.

2. 움직임 벡터 부호화

Inter 예측에서는 참조 영상에서 현재 블록과 가장 유사한 블록을 찾기 위하여 움직임 예측을 수행하며, 예측블록과 현재 블록의 차이 신호를 변환 및 양자화를 통해 부호화한다. 두 블록간의 상대적인 위치 정보를 움직임 벡터로 표현하고, 이 값을 주변블록의 움직임 벡터로 예측하여 차이신호만을 전송한다. HEVC의 참조 소프트웨어인 HM6.0에서 최적의 움직임 벡터를 찾기 위해 다음과 같은 비용함수^[2]를 사용한다.

$$J_M = D + \lambda R(MVD) \quad (1)$$

수식 (1)에서 D 는 현재 블록과 예측블록의 예측 오차이며, $\lambda R(MVD)$ 는 움직임 벡터와 예측 움직임 벡터의 차분 움직임 벡터의

1) 연락처: 한종기

본 연구는 지식경제부 및 정보통신산업진흥원의 IT융합 고급인력과정 지원사업의 연구결과로 수행되었음
(NIPA-2012-H0401-12-1003)

비트량을 나타낸다. HEVC에서는 D 를 계산하기 위하여 다음과 같은 Sum of Absolute Difference(SAD) 왜곡 측정 방법을 사용한다.

$$D = \sum_{i=0}^{H-1} \sum_{j=0}^{W-1} |O(i, j) - P(i, j)| \quad (2)$$

수식 (2)에서 O 와 P 는 현재 블록과 예측블록의 픽셀값을 나타낸다. H 와 W 는 현재 블록의 가로와 세로는 나타낸다. 수식 (1)에서 비트량 R 은 다음과 같은 수식으로 계산한다.

$$R = Exp(MVD) \quad (3)$$

수식 (3)에서 $Exp(X)$ 는 입력값 X 를 부호화 하는데 필요한 비트량을 지수 곱셈 코드로 계산하는 함수이다. MVD 는 차분 움직임 벡터를 나타내며 다음과 같이 표현할 수 있다.

$$MVD = MV - MVP \quad (4)$$

그리고 MVD 를 표현하기 위해 변형된 1차 곱셈 코드를 사용한다.

MVD	비트 스트링
0	0
1	10
2	1110
3	1111
4	110100
5	110101
6	110110
7	110111

표 1. 차분 움직임 벡터를 부호화 하는 지수 곱셈 코드

그림1 은 HEVC에서 현재 PU의 MVP의 후보를 나타낸 그림이다.

MVP의 집합 $\{Am, Am+1, Bn-1, Bn, Bn+1\}$ 중에서 2개의 MVP를 선택한다. 두 개 중의 가장 현재 움직임 벡터와 비슷한 MVP를 선택한다. 두 개 중의 선택된 MVP는 flag를 디코더에 전송하여 어떤 것이 선택되었는지 구분한다.

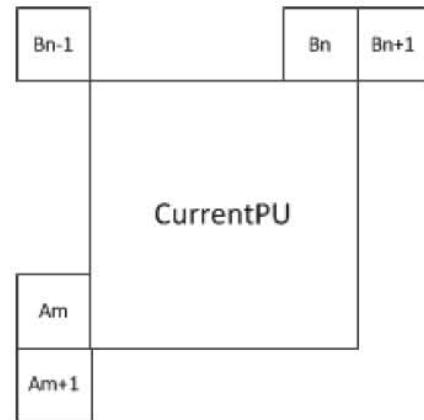


그림 1. MVP의 후보

서론에서 서술하였다시피 낮은 차수를 이용하여 MVD를 부호화 하면 작은 MVD에 짧은 비트를 할당하기 때문에 차분 움직임 벡터가 작을 때 성능 향상을 얻을 수 있다. 하지만 차분 움직임 벡터가 큰 경우 낮은 지수 곱셈 코드로 부호화 한다면 높은 지수 곱셈 코드를 사용하였을 때 보다 부호화 효율이 낮아지게 된다.

3. 차분 움직임 벡터 코딩을 위한 적응적인 지수 곱셈 코드 차수 선택 방법

차분 움직임 벡터가 매우 커서 1차 지수 곱셈 코드로 부호화 하기 어렵다면 차분 움직임 벡터에 알맞은 지수 곱셈 코드를 선택하여 정확도가 높은 예측블록을 만들 수 있을 뿐만 아니라 차분 움직임 벡터의 비트 량도 줄일 수 있다. 현재블록의 움직임 추정을 하기 전에 주변블록들의 MV를 이용하여 현재 MV를 예측한다.

예측 움직임 벡터가 실제 발생한 움직임 벡터와 유사하면 차분 움직임 벡터는 매우 작기 때문에 낮은 차수의 지수 곱셈 코드를 할당한다. 예측 움직임 벡터가 실제 발생한 움직임 벡터와 다르면 차분 움직임 벡터는 높은 차수의 지수 곱셈 코드를 할당한다. 선택된 MVP를 알 수 있으면 현재블록의 발생할 MV를 예측할 수 있는데 다음 그림 2에서는 Racehorses를 실제로 코딩하여 나온 MVP와 MV의 관계를 나타낸 그림이다.

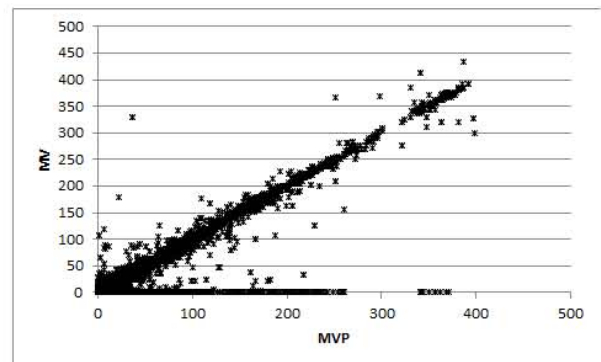


그림 2. Racehorses의 MVP와 MV의 발생

이 차트에서 MVD와 MV의 관계를 다음 수식으로 나타낼 수 있다.

$$MV_p = \alpha MVP + \beta \quad (5)$$

수식 (5)에서 $\alpha = 0.8914$, $\beta = -1.2638$ 의 값으로 넣어 줄 수 있는데 이 값들은 모델링을 통하여 구할 수 있다. 모델링으로 α 와 β 값을 알아 낼 수 있으므로 이제 두 개의 MVP 에서 어떤 MVP 가 선택 될 것인지를 움직임 추정 전에 미리 계산하여야 한다. 현재 PU의 복호화된 주변 화소 값들을 PU의 왼쪽에 있으면 LPU, 위에 있으면 APU 라고 정의한다. MVP₁ 으로 LPU와 참조블록의 D 를 구하고 MVP₂ 을 APU와 참조블록의 D 를 구하여 더 작은 D 를 가지는 MVP 을 현재 블록의 MVP 로 정한다.

위에서 구한 MVP 를 수식 (5)에 대입하면 MV_p 를 구할 수 있다. 위에서 구한 MV_p 와 MVP 를 알면 현재 PU의 차분 움직임 벡터를 계산할 수 있다.

$$MVD = MV_p - MVP \quad (6)$$

$$Min\{\exp G_i(MVD)\}, i = 0, 1, 2, 3$$

수식 (6)에서 i는 지수 곱셈 코드의 차수를 이야기 한다. 수식 (6)을 만족하는 지수 곱셈 코드의 차수를 계산하여 움직임 추정을 수행하여 부호화 한다.

4. 실험 결과 및 분석

4장에서는 제안하는 알고리즘을 적용한 실험결과를 보인다. 실험에서는 HEVC의 참조 소프트웨어인 HM6.0^[3]을 사용하였으며 JCT-VC에서 권고하는 일반 실험 조건을 적용하여 제안하는 알고리즘을 검증하였다^[4]. 실험결과에서 부호화 효율은 BD-Rate를 사용하여 측정하였으며, 결과가 음수 값을 가지면, 비교 대상과 비교해서 부호화 효율이 향상되었음을 의미하고, 양수 값을 가지면 부호화 효율이 저하되었음을 나타낸다.

Class D	BQ Square	-0.1%	0.0%	-0.4%
	Race Horses	-0.6%	-2.1%	-2.0%
Class F	Basketball Drill Text	-0.3%	-1.0%	-1.3%
	China Speed	-0.8%	-1.5%	-1.6%
	Slide Show	-0.3%	-1.7%	-2.1%

5. 결론

본 논문에서는 현재 블록의 MVP로 지수 곱셈 코드의 차수를 결정하는 기술을 제안하였다. 제안하는 알고리즘은 예측 MVD값이 크면 높은 차수의 지수 곱셈 코드를 사용하고 예측 MVD값이 작으면 낮은 차수의 지수 곱셈 코드를 사용하였다. 실험을 통하여 제안하는 알고리즘이 HM6.0과 비교하여 Random Access 부호화 구조에서는 평균적으로 약 0.4%의 성능 향상을 얻을 수 있었다.

참고문헌

- [1] The Joint Collaborative Team on Video Coding (JCT-VC) of TU-T WP3/16 and ISO/IEC JTC 1/ SC29/ WG11 held its eighth meeting during 1-10 Feb 2012 at the DoubleTree Inn in San Jose, US.
- [2] K. McCann, B. Bross, I-K. Kim, W-J. Han, "HM6: High Efficiency Video Coding (HEVC) Test Model 6 Encoder Description", JCTVC-H1002. San José, CA, USA, 1-10 February, 2012
- [3] <http://hevc.kw.bbc.co.uk/trac/browser/jctvc-hm/tags/HM-6.0>
- [4] F. Bpsen, "Common test condition and software reference configurations", JCTVC-H1100, San José, CA, USA, 1-10 February, 2012