특집(레터)논문-11-16-5-10 http://doi.org/10.5909/JEB.2011.16.5.793

H.264/AVC 동영상 압축율 향상을 위한 DC 오프셋 보정에 기반한 인터 예측 알고리즘

윤 대 일[a], 김 해 광[a]‡

DC Offset Adjusted Inter Prediction Algorithm for Improving H.264/AVC Video Coding Efficiency

Dae Il Yoon[a] and Hae Kwang Kim[a]‡

요 약

H.264/AVC 동영상 압축 표준 기술은 인터/인트라 예측에 의한 잔여 영상을 주파수 변환하고 양자화하여 엔트로피 부호화한다. 이 논문에서는 인터 예측에 의한 잔여 영상을 주변 영상 블록의 영상 정보를 사용하는 기존의 방법을 향상시킨 DC 오프셋 보정 방법에 의해 잔여 영상의 정보량을 감소하는 것에 의해 압축 성능을 향상시키는 기술을 제안한다. DC 오프셋에 관련한 오버헤드 정보는 비트스트림에 포함하지 않고 인코더와 디코더가 같은 방법의 DC 오프셋을 수행한다. 실험결과는 H.264/AVC에 비해서는 BD-Rate으로 평균 0.25% 좋아지지만 기존의 DC 오프셋 방법과 비교해서는 영상 시퀀스에 따라 성능이 향상 혹은 저하되는 것을 보여 준다. 평균적으로는 0.09%의 압축 성능이 저하된다. 이 논문의 실험결과를 통해 기존의 DC 오프셋 방법과 제안하는 방법을 슬라이스 단위, 매크로블록 단위 등의 부호화단위에 따라 적응적으로 적용하는 방법에 의한 압축율 향상에 대한 가능성을 확인하였다.

Abstract

H.264/AVC compresses video data by applying DCT transform, quantization and entropy coding processes to the residual signal obtained by inter/intra prediction. This paper proposes a method enhancing an existing DC offset adjustment technology which uses information of neighboring blocks to reduce residual information for improving coding efficiency. DC offset information is not sent over bitstreams, but calculated in the same way both in the decoder and in the encoder. Experimental results show that the proposed method enhances coding efficiency by 0.25% in average BD-Rate compared to H.264/AVC and gives better or worse coding efficiency compared to the existing DC offset method depending on video sequences with coding efficiency degradation by 0.09% in average BD-Rate. This experimental results also show that further coding efficiency improvement is possible by applying the proposed method adaptively to slice or macroblock coding units.

Keywords : Video compression, DC offset, Coding efficiency, H.264/AVC

Ⅰ. 서 론

H.264/AVC 동영상 압축 기술을 포함한 대부분의 동영상 압축 기술들은 인트라 혹은 인터 예측에 의해 잔여 영상

a) 세종대학교 컴퓨터공학과
Department of Computer Engineering, Sejong Unsiversity
‡ 교신저자 : 김해광(hkkim@sejong.ac.kr)
※ 이 논문은 2009년도 세종대학교 교내연구비 지원에 의한 논문임.
・접수일(2011년7월14일),수정일(2011년8월29일),게재확정일(2011년8월29일)

을 부호화하는 방법[1]에 의해 압축 성능을 높인다. 인터 예측의 성능을 향상시키기 위해서는 서브펠(sub-pel)에 기반한 정밀 움직임 벡터 해상도를 위한 인터폴레이션 방법들과 밝기 보정에 의한 인터 예측 향상 방법들[2]이 제안되어 왔다. H.264/AVC에서는 밝기 보정에 의한 인터 예측 향상 방법으로서 가중치 예측(weighted prediction) 방법[3]이 채택되었다. 가중치 예측 방법은 식 (1)과 같이 움직임 예측에 의한 참조 블록의 픽셀에 스케일성분 a를 곱하고 오프셋값 b을 더하여 예측 블록을 만든다. H.264/AVC에서 스케일성분 a 와 오프셋값 b의 값은 슬라이스 단위로 계산되어 비트스트림에 포함하여 디코더에 전송한다.

$$I(x,y,t) = a \bullet I(x+\Delta x, y+\Delta y, t-\Delta t) + b \quad (1)$$

H.264/AVC의 가중치 예측 방법에서 스케일성분 a와 오프셋 값 b의 계산 방법은 인코더 방법으로서 표준 사항은 아니다. 간단한 방법으로서는 현재 픽쳐의 평균값과 참조 픽쳐의 평균 값을 사용하여 스케일성분 a를 구하고 스케일된 참조 픽쳐를 사용하여 최적의 오프셋값 b를 구할 수 있다. 가중치 예측 방법은 페이드(fade)와 같이 전체 화면의 밝기가 고르게 변화하는 영상에 적합하다.

H.264/AVC의 다중시점 부호화 표준에서는 매크로블록 단위로 현재 블록과 참조 블록의 최적 DC 오프셋을 계산하고 부호화하여 비트스트림에 포함하는 방법[4]이 제안되었다. 이 방법은 다중 시점의 영상 사이에서 지역적 밝기 변화에 대해 압축율 향상 효과를 실험결과로 보여준다.

블록 단위로 DC 오프셋을 주변 영상 블록을 참조하여 유도하여 예측하는 방법[5]이 연구되었다. 이 방법은 DC 오프셋 값을 부호화하지 않고 부호화할 블록의 DC 오프셋 값을 부호화할 블록의 이미 부호화된 이웃 블록들과 이 이웃 블록들이 각각 참조하고 있는 참조 블록들 사이의 SD (Sum of Difference)의 픽셀 평균값으로 DC 오프셋 값을 유도한다. 이 방법에서는 DC 오프셋을 유도할 때 현재 블록의 움직임 예측에서 사용한 같은 참조 프레임을 사용하지 않는 이웃 블록들은 제외하는 방법을 제안하고 있다.

II. 제안하는 DC 오프셋 유도 방법

제안하는 DC 오프셋 유도 방법은 기존의 DC 오프셋 유도 방법과 달리 상이한 참조 프레임만이 아니라 현재 블록의 움직임 벡터와 유사한 움직임 벡터를 갖지 않는 이웃 블록들은 DC 오프셋 유도에서 제외한다. 그 이유는 현재 블록과 참조 블록사이의 밝기 변화는 공간적으로 인접한 이웃 블록에서 유사한 값을 가질 수 있으나 이웃 블록의 참조 블록이 다른 참조 프레임에 속하거나 혹은 같은 참조 프레임에 있더라도 현재 블록의 참조 블록과의 위치가 먼 경우에는 밝기 변화의 상관성이 적어지기 때문이다.

그림 1은 이 논문에서 제안하는 방법에 따른 이웃 블록 집합의 구성의 예를 보여준다.

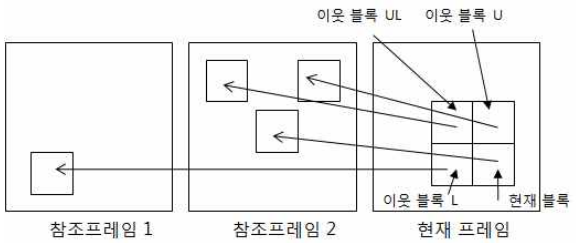

그림 1. 이웃블록집합의 구성
Fig. 1. Composition of neighboring blocks

그림 1에서 현재 블록에 대한 이웃 블록 후보 집합은 왼쪽 이웃 블록 L, 위 이웃 블록 U, 위-왼쪽 이웃 블록 UL을 사용하여 {L, U, UL}로서 구성된다. 현재 블록과 이웃 블록은 움직임 예측에 따른 참조 프레임 색인과 움직임 벡터에 의해 화살표가 표시하는 참조 블록을 가지고 있다. 그림 1의 예에서 이 논문에서 제안하는 방법에 따르면 이웃 블록 L은 현재 블록과 다른 참조 프레임을 참조하고 이웃 블록 UL은 참조 프레임은 같지만 현재 블록의 움직임 벡터와 유사하지 않기 때문에 DC 오프셋을 계산하기 위한 이웃 블록 집합에 포함하지 않는다. 움직임 벡터의 유사성은 식 (2)를 사용하여 판단한다. 식 (2)에서 MVCx는 현재 움직임 벡터의 x 성분, MVCy는 현재 움직임 벡터의 y 성분, MVNx는 이웃 움직임 벡터의 x 성분, MVNy는 이웃 움직임 벡터의 y 성분을 나타낸다. 움직임 벡터의 단위는 1/4 화소 단위로 가장 유사한 이웃 블록들을 포함시키기 위하여 7로 임계값

을 설정하여 7보다 크면 이웃 블록 집합에 포함시키지 않는다. 임계값 7은 실험을 통해 압축율 향상이 최적인 값으로 설정하였다.

$$|MVCx - MVNx| + |MVCy - MVNy| > 7 \quad (2)$$

MxN 크기의 블록에 대해 이웃 블록 집합에 속하는 이웃 블록 i가 l개 있는 경우 DC 오프셋의 유도 공식은 식 (3)와 같다. 이웃블록 i에 대해 식 (4)로 SD를 구한다.

$$DC\text{오프셋} = \frac{1}{M \times N \times l} \sum_{i \in \text{이웃블럭집합}} SD_i \quad (3)$$

$$SD_i = \sum_{m,n} OrignialBlock(m,n) - ReconBlock(m,n) \quad (4)$$

예측 보정에는 식 (3)에서 얻은 DC 오프셋 값을 [-2, 2]의 구간의 정수로 클리핑하여 사용한다. 이 구간의 사용은 기존의 DC 오프셋 유도 방법[5]에서 제시한 실험 결과에 의하면 평균적으로 약 80%의 매크로블록에 있어서 잔여 영상의 평균픽셀 값이 [-2, 2] 구간에 있기 때문이다.

제안하는 DC 오프셋 유도 방법의 알고리즘은 다음과 같다.

DC오프셋 유도 알고리즘

시작

- 현재 블록의 움직임 예측에 의한 참조 프레임 색인과 움직임 벡터 결정
- 이웃 블록 후보 집합에서 참조 프레임 색인이 다르거나 움직임 벡터가 유사하지 않는 이웃 블록들은 제외하여 이웃 블록 집합 생성
- 이웃 블록 집합들의 이웃 블록들과 해당하는 참조 블록들의 DC 오프셋을 계산
- DC 오프셋을 [-2, +2] 구간으로 클리핑하여 최종 DC 오프셋 값 결정
- 현재 블록의 참조 블록에 최종 DC 오프셋을 더하여 예측 블록 생성
- 현재 블록과 예측 블록으로부터 잔여 영상 블록 생성

끝

III. 실험 결과

제안하는 방법은 JM15.1 기반에 구현하고 기존의 DC 오프셋 유도 방법 및 JM15.1과 압축 성능을 비교하였다. 표 1은 제안하는 방법의 실험 환경 설정을 나타낸다.

표 1. 실험환경 설정
Table 1. experimental configuration setting

YUV format	YUV 4:2:0
Prediction Structure	IPPP
Number of reference frames	4
Entropy coding	CAVLC
RDOptimization	high-performance mode
Motion vector search range	64
RDOptimizedQuantization	ON

표 2는 제안하는 DC 오프셋, 기존의 DC 오프셋[5], JM15.1의 비교 실험 결과를 보여준다. QP값 37, 32, 27, 22를 파라미터로 가지고 HD(1280x720), CIF(352x288), QCIF(176x144) 크기 시퀀스들의 실험 결과를 보여준다.

제안하는 DC 오프셋은 H.264/AVC에 비해 평균 BD-Rate은 0.25% 향상되었으며 기존의 DC 오프셋에 비해 평균 BD-Rate은 0.09% 저하되었다.

표 2. 제안하는 DC 오프셋의 비교 실험 결과
Table 2. comparative experimental results of proposed DC offset

Sequences		QP	제안한 DC 오프셋		기존의 DC 오프셋		JM15.1	
			PSNR(Y)	Total Bits	PSNR(Y)	Total Bits	PSNR(Y)	Total Bits
1280 x 720	Shuttle Start	37	36.88	555112	36.89	553432	36.87	555664
		32	38.72	1105440	38.74	1105288	38.68	1114792
		27	41.17	2341944	41.16	2354832	41.13	2377704
		22	43.49	6330344	43.50	6332264	43.49	6327064
		BD-Rate / BD-PSNR			-0.028 / 0.005		-1.987 / 0.601	
352 x 288	tempete	37	27.70	890136	27.67	887512	27.68	889808
		32	31.05	2147888	31.06	2144296	31.06	2137960
		27	35.00	5549144	35.00	5546112	34.99	5522832
		22	39.28	11971832	39.27	11977720	39.26	11936936
		BD-Rate / BD-PSNR			0.076 / -0.004		0.285 / -0.014	
	paris	37	28.60	628304	28.58	629072	28.59	628096
		32	32.14	1295560	32.12	1296048	32.12	1297888
		27	35.95	2638872	35.95	2639400	35.95	2637000
		22	39.70	4930568	39.71	4936792	39.70	4924152
		BD-Rate / BD-PSNR			-0.225 / 0.012		-0.186 / 0.011	
	mobile	37	26.55	1326928	26.58	1335728	26.56	1328216
		32	30.19	3382088	30.19	3380376	30.19	3369776
		27	34.42	8646448	34.42	8643672	34.41	8607520
		22	38.97	17522504	38.97	17510896	38.96	17455144
		BD-Rate / BD-PSNR			0.055 / -0.002		0.283 / -0.014	
	foreman	37	31.29	516808	31.35	515392	31.29	517432
		32	34.11	917928	34.10	917448	34.06	923728
		27	36.98	1767152	36.98	1765872	36.97	1778928
		22	40.08	3769016	40.09	3771608	40.09	3789856
		BD-Rate / BD-PSNR			0.171 / -0.009		-1.001 / 0.043	

176 x 144	silent	37	29.60	109176	29.65	109128	29.67	109344
		32	32.91	211224	32.93	209928	32.96	210568
		27	36.51	392104	36.55	392448	36.55	391072
		22	40.52	717048	40.53	718248	40.53	714352
		BD-Rate / BD-PSNR			0.678 / -0.039		1.021 / -0.059	
	news	37	29.98	113608	30.00	113184	30.03	114104
		32	33.52	207168	33.53	205800	33.53	205936
		27	37.34	379832	37.35	381280	37.34	378296
		22	40.94	693016	40.96	692912	40.96	692360
		BD-Rate / BD-PSNR			0.350 / -0.021		0.546 / -0.033	
	foreman	37	30.08	190296	30.03	190320	30.07	191672
		32	33.41	334200	33.37	332664	33.38	336056
		27	36.72	602312	36.70	601224	36.69	603080
		22	40.14	1134040	40.15	1132952	40.14	1134256
		BD-Rate / BD-PSNR			-0.217 / 0.013		-0.758 / 0.043	
	container	37	30.08	48072	30.08	48072	30.07	48536
		32	33.39	83256	33.39	83128	33.40	84560
		27	36.53	181592	36.53	181976	36.55	181592
		22	39.84	473624	39.84	474768	39.85	475872
		BD-Rate / BD-PSNR			-0.068 / 0.004		-0.465 / 0.015	

Paris와 ShuttleStart, Container의 경우 각각 0.23%, 0.03%, 0.07%의 비트율 감소를 보여준다. Foreman의 176x144의 경우 -0.22% 비트율 감소를 보여주나, 352x288의 경우 오히려 0.17%의 비트율 증가를 보여준다. Silent와 News, Tempete, Mobile 시퀀스에서는 비트율 증가가 관찰된다.

표 3은 QP32를 파라미터로 준 Paris 시퀀스의 각 모드들의 빈도수를 보여준다.

표 3. Paris QP32에서 각 모드의 빈도수
Table 3. counts of coding modes with Paris QP 32

Paris QP32	제안한 DC 오프셋	기존의 DC 오프셋	JM15.1
Mode 0 (copy)	40704	40683	40754
Mode 1 (16x16)	6763	6785	6657
Mode 2 (16x8)	2549	2636	2676
Mode 3 (8x16)	2963	2893	2876
Mode 4 (8x8)	5677	5677	5708
Mode 5 intra 4x4	127	134	131
Mode 6 intra 8x8	51	41	52
Mode 7+ intra 16x16	170	155	150
Mode intra IPCM	0	0	0

제안하는 DC 오프셋은 기존의 DC 오프셋, JM15.1보다 이웃 블록들의 유사한 움직임을 가지고 잔여 값을 줄여서 DC 오프셋, JM15.1보다 1 비트만으로 부호화가 가능한 스킵 모드의 선택이 많아져서 압축율 향상을 보여준다. 비트수가 많은 인트라 모드의 수는 DC오프셋과 JM15.1 보다 늘어났지만 총 예측블록 중 차지하는 비율이 적어서 전체적으로는 압축율 향상을 보여준다.

실험결과에 따르면 기존의 DC 오프셋이 전반적으로 H.264/AVC에 비해 압축율 향상을 보이지만, 시퀀스와 QP에 따라 일부 압축율 저하 현상이 나타나며, 제안하는 방법도 시퀀스와 QP에 따라 기존의 DC 오프셋과 비교하여 압축성능이 상이하게 나타나는 현상이 나타난다. 기존의 DC 오프셋 방법과 제안하는 방법을 슬라이스 단위, 매크로블록 단위 등의 부호화단위에 따라 오버헤드 비트를 비트스트림에 기록하여 적응적으로 적용하는 방법에 의한 압축율 향상이 기대된다.

VI. 결론 및 향후 연구방향

이 논문에서는 인터 예측에 의한 잔여 영상을 주변 영상 블록의 영상정보를 사용하는 기존의 방법에서 참조 프레임이 다르거나 혹은 참조 프레임이 같더라도 움직임 벡터가 유사하지 않는 이웃 블록들은 DC 오프셋 유도에서 제한하여 성능을 향상시키는 방법을 제안하였다. 실험을 통하여 제안한 방법이 H.264/AVC에 비해 평균 0.25% 압축율 향상 기존의 DC 오프셋 방법보다 평균 0.09%의 압축율 성능 저하를 관찰하였다. 시퀀스와 QP에 따라 기존 방법과 제안하는 방법의 압축 성능이 상이하게 나타난다. 이를 해결하는 향후 연구 방향으로서 기존의 방법과 제안하는 방법을 슬라이스 단위, 매크로블록 단위 등의 부호화단위에 따라 오버헤드 비트를 비트스트림에 기록하여 적응적으로 적용하는 방법에 의한 연구를 계획하고 있다.

참 고 문 헌

[1] ITU-T and ISO/IEC JTC 1, "Advanced Video Coding for Generic Audio-Visual Services. ITU-T Recommendation H.264 and ISO/IEC 14496-10 (MPEG4-AVC), Version 4," July 2005

[2] T.Wedi, "1/8-pel motion vector resolution for h.261," ITU-T Q.15/SG16 Doc. Q15-K-21, 2000

[3] J.Boyce, "Weighted prediction in the H.264/MPEG AVC video coding standard," Proc. IEEE ISCAS, pp789-792, Vancouver, Canada, 2004

[4] J. Hur, S. Cho and Y. LEE, "Adaptive Local Illumination Change Compensation Method for H.264/AVC-Based Multiview Video Coding," IEEE Trans. on CSVT, vol.17, no. 11, pp1496-1505, 200

[5] Jie JIA, Daeil Yoon and Hae Kwang Kim, "A predictive block based DC offset for H.264/AVC video coding", IEICE trans. on Fundamentals, Vol. 93-A(5): pp976-980, 2010