새로운 구문 요소를 이용한 개선된 H.264 동영상 압축 방법

정회원 백 성 학*, 문 용 호**, 김 재 호*

An improved H.264 Video Encoding using a New Syntax Element

Seong-Hak Baek*, Yong-Ho Moon**, Jae-Ho Kim* *Regular Members*

요 약

본 논문에서는 H.264의 동영상 압축 성능 향상을 위하여 새로운 구문 요소를 제안한다. 기존 H.264 압축 부호화 방식에서는 RUN, MB_type이라는 독립된 구문 요소로서 매크로 블록 정보를 전송한다. 그런데 다양한 테스트 시퀀스들에 있어서 RUN, MB_type 구문 요소들을 하나로 결합할 경우 압축 성능이 보다 향상된다는 사실을 실험적으로 확인하였다. 이와 같은 사실을 바탕으로, 본 논문에서는 기존의 독립된 2가지 구문 요소를 하나로 통합한 새로운 (RUN, MB_type) 구문 요소를 제안한다. 그리고 보다 성능 개선을 위하여 제안한 구문 요소의 부호로서 화면 종료를 나타내는 EOF 부호를 새로이 추가하여 사용하였다. 모의실험 결과에서는 기존 방식에 비해 제안한 방식이 매크로 블록의 압축률에 있어 최대 15% 향상됨을 보여준다.

Key Words : Video compression coding, Syntax element

ABSTRACT

In the H.264 video coding standard, the information for the macroblock is encoded by the independent syntax elements, called RUN and MB_type. Through the experimental observation for various test sequences, it is shown that the performance of the video coding is improved by using the combined RUN and MB_type as an entity. Based on this fact, we propose an unified new syntax element composed of the RUN and MB_type in order to improve the compression ratio of H.264 video coding. In the proposed syntax element, an EOF(End-Of-Frame) symbol indicating the end of picture is used for enhancing the performance of video coding. The simulation result shows that the proposed method outperforms maximally 15 % to the conventional method for the bitrate of the macroblock.

I. 서 론

현재 인터넷의 대중화, 향상된 네트워크 기술, 그리고 통신기기 등의 발달로 유·무선 정보 통신망을 이용한 멀티미디어 동영상 서비스에 대한 관심은 날로 증가되고 있는 추세이다. 지금까지 효율적인 동영상 압축을 위하여 MPEG-1, 2, 4, H.26x 등과 같은 다양한 표준안들이 제정되어 왔다. 한편 보다 높은 압축률의 요구로 VCEG (Video Coding Experts Group - ITU-T SG16 Q.6)에서는 1998년

* 부산대학교 전자공학과 영상통신연구실({baeksh, jhkim}@pusan.ac.kr), ** 부산외국어대학교 (yhmoon5@taejo.pufs.ac.kr)
논문번호 : 030190-0509, 접수일자 : 2003년 5월 2일

H.26L이라고 하는 명칭으로 새로운 압축 표준화 작업을 시작하였다. 이 표준화의 목표는 현재 기존의 동영상 압축 표준안에 비해 2배의 압축 성능을 갖는 부호화 방식을 제정하는 것이다. 그 이후 2001년 MPEG(Moving Pictures Expert Group - ISO/IEC JTC 1/SC 29/WG 11)에서는 H.26L의 성과를 인식하고, VCEG와 합쳐 JVT (Joint Video Team)를 결성하였다. JVT의 주 목적은 H.26L를 초안으로 새로운 국제 표준안의 공식 명인 H.264/AVC (Advan ced Video Coding)으로 제정하는 것으로, 주요 특징은 다음과 같다[1].

- Directional spatial prediction for intra coding.
- Variable block-size motion compensation with small block sizes.
- Multiple refernce picture motion compensation.
- small block-size integer transform
- Context-adaptive entropy coding (:CAVLC)
- Arithmetic entropy coding (:CABAC)
- In-the-loop deblocking filtering

이와 같은 두드러진 특징으로 H.264는 기존의 동영상 압축 표준안보다 압축 성능이 훨씬 우수한 것으로 평가되고 있다[2].

특히 H.264에서는 기존의 표준안과는 달리 매크로 블록(MB:MacroBlock, 16x16)을 다양한 일정 크기와 모양으로 분할한 부블록 기반의 화면 내/간 예측 부호화한다. 이는 매크로 블록에 대한 추가적인 움직임 벡터와 예측 모드 정보(MB_type)가 요구된다. 한편 양자화된 예측 오차 신호 등과 같은 일반적인 매크로 블록의 정보 없이도 복원 가능한 매크로 블록으로서 생략 매크로 블록이 있다. 이와 같이 생략 매크로 블록에 대한 정보(RUN)는 필요에 의해 매크로 블록 정보들이 전송될 때 연속 길이 부호화로 부호화되어 전송된다.

H.264에서는 매크로 블록 정보로서, 양자화된 예측 신호를 제외한 앞서 언급한 MB_type과 RUN 구문 요소를 포함한 모든 구문 요소(움직임 벡터, CBP 등)를 각각 독립적으로 다루어 특정 Exp-Golomb 가변 길이 부호어로 부호화한다[3]. 그러나 이와 같은 엔트로피 부호화 방식은 모든 구문 요소들을 독립적으로 다루어 부호화하므로 구문 요소간의 상관 관계는 전혀 고려하지 않는다.

본 논문에서는 이와 같은 매크로 블록층 구문 요소 중에서 RUN 구문 요소에 대한 문제점을 인식하고, 이를 해결하기 위해 다른 구문 요소들과의 상관성 여부를 파악하였다. 그리고 RUN 구문 요소와 MB_type 구문 요소사이에는 상호 상관성이 있음을 실험적 사실을 통한 엔트로피(entropy) 관점에서 확인하였다. 이를 토대로, 기존의 독립된 두 구문 요소 RUN, MB_type를 쌍으로 하는 새로운 하나의 (RUN, MB_type) 구문 요소로 다룰 것을 제시한다. 한편 H.264에서는 경우에 따라 부호화하는 화면마다 발생되는 화면 종료 신호를 추가적으로 RUN 구문 요소로 부호화하여 전송한다. 그러나 이들 신호들을 하나의 EOF(End-Of -Frame) 부호로 다룬다면 개별적으로 다루는 것보다는 효율적이라는 사실도 실험적으로 확인하였다. 결론적으로 보다 개선된 부호화 효율을 위하여 앞서 제안한 (RUN, MB_type) 구문 요소에 EOF 부호를 새로이 추가한, 보다 효과적인 {(RUN, MB_type)+EOF} 형태의 구문 요소를 제안한다. 그리고 제안한 구문 요소도 실험적 사실을 통한 엔트로피 관점에서 검증하였다.

본 논문의 구성은 Ⅱ장에서는 제안하는 알고리즘에 대한 이해를 돕기 위해 H.264 부호화 방식에 대하여 간단히 설명한다. 그리고 Ⅲ장에서는 제안하는 알고리즘을, Ⅳ장에서는 제안한 알고리즘에 대한 성능 평가와 Ⅴ장에서 결론을 맺는다.

Ⅱ. H.264 부호화 방식의 이해

여기서는 그림 1과 같은 H.264 동영상 압축 부호화에서 예측 부호화와 엔트로피 부호화에 대하여 간략하게 설명한다[1]. 일반적으로 예측 부호화의 기본 개념은 연속되는 화소들 사이의 시/공간적 중복성을 제거하고 새로운 정보만을 부호화하는 것이다. 이때 화소에 대한 예측치는 크게 화면 내 예측과 화면 간 예측에 의해 얻어진다. 그리고 엔트로피 부호화는 통계적 중복성을 제거하여 부호화하는 것으로, 보통 가변 길이 부호화(VLC)가 널리 사용된다.

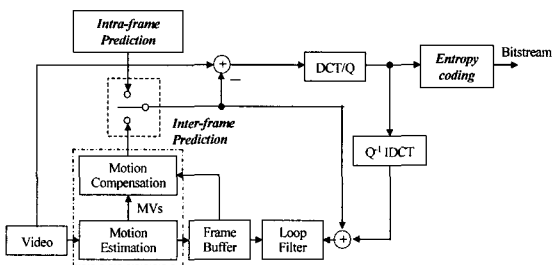

그림 1. H.264 동영상 압축 부호화 블록도

1. 예측 부호화

화면 내 예측 부호화는 한 화면 내 화소에 대한 예측치를 인접한 화소간의 공간적 상관성을 이용하여 구하고, 이렇게 구한 예측치를 부호화한다. H.264에서는 블록 기반의 DC 또는 하나 이상의 방향에 따라 INTRA-4×4와 INTRA-16×16 예측 모드로 예측치를 구한다[1].

화면 간 예측 부호화는 연속되는 화면간의 시간적 상관성을 이용하여 현재 화면의 화소에 대한 예측치를 이전 화면으로부터 구하고, 이렇게 얻은 예측 오차를 부호화한다. 이는 보편적으로 이전 화면으로부터 예측치를 구하기 위해서 화면 간의 움직임을 구하는 "움직임 추정"과, 이렇게 추정한 움직임으로부터 예측치 및 그 예측 오차를 구하는 "움직임 보상"에 의해 이루어진다. 현재 기존 표준안에서는 16×16 또는 8×8 블록 내의 모든 화소들에 대하여 하나의 움직임 벡터로 구하고 보상하는 블록 정합 알고리즘(BMA)을 사용한다. H.264에서는 매크로 블록을 그림 2와 같이 일정한 크기와 모양으로 최소 4×4 부블록까지 분할하고, 그 부블록 내의 모든 화소들에 대하여 화면 간 예측 부호화한다.

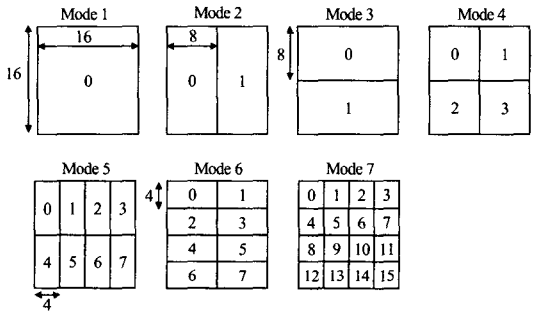

그림2. H.264 화면 간 예측 모드

이는 부호화할 매크로 블록당 최대 16개의 움직임 벡터 정보가 추가적으로 요구된다. 그리고 앞서 언급한 화면 내/간 예측 모드 정보(MB_type)도 부가적으로 요구된다. 또한 이렇게 구한 움직임 보상 블록 내 화소에 대한 예측치는 다수의 이전 화면들로부터 구하므로, 이에 대한 참조된 화면의 색인 정보(Ref_frame)도 요구된다. 한편 이외에도 양자화된 예측 오차 신호 등과 같은 일반적인 매크로 블록 정보 없이도 복원이 가능한, 이른바 생략 매크로 블록이 있다. 그러므로 이와 같은 특정 매크로 블록에 대한 정보도 요구되는데, 이는 매크로 블록 정보들이 필요에 의해 전송될 때 이전에 존재했던 생략 매크로 블록의 연속 길이(RUN)로서 부호화하여 전송된다. 이와 같이 H.264에서의 부가적인 매크로 블록 정보들은 그림 3과 같이 양자화된 예측 오차 신호(Tcoeff)를 제외한 모든 매크로 블록층 구문 요소와 더불어 각각 독립적으로 다루어 Exp-Golomb 부호어라고 하는 특정 부호어로 가변 길이 부호화하여 최종적으로 전송된다.

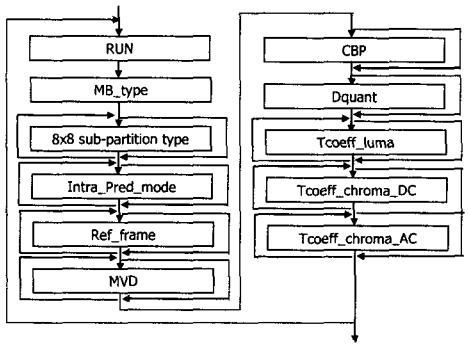

그림3. H.264 매크로 블록층 구문 요소

2. 엔트로피 부호화

H.264 엔트로피 부호화에서는 양자화된 예측 오차 신호를 CAVLC(Context-Adaptive VLC)로, 이외 모든 구문 요소는 Exp-Golomb 부호어로 부호화하는 것과 주위 문맥 정보를 이용하여 부호화하는 CABAC (Context-Adaptive Binary Arithmetic Cod -ing)[6]으로 나뉜다. 특히 Exp-Golomb 부호화에서는 그림 4와 같이 정규화된 대칭적 구조의 부호어 형태를 지닌다.

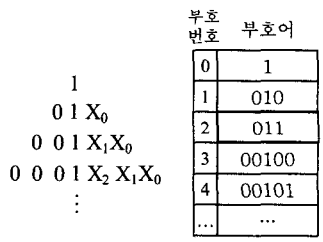

그림4. H.264 Exp-Golomb 부호어 구조 형태

이와 같은 Exp-Golomb 부호어 구조 형태는 정적 확률 분포 특성을 지니며, 그 구조는 식 (1)과 같이 *[M 0s]*와 *[INFO]* 영역으로 크게 구분된다.

[M 0s][1][INFO] (1)

여기서, *[M 0s]* 영역은 *M*-비트의 '0(zero)'이 수반되는 기본 코드 영역이고, *[INFO]* 영역은 *M*-비트 영역을 수반하는 실제적인 정보(=$X_n...X_1X_0$) 코드 영역이다. 그러므로 전체 부호어 길이는 (2*M*+1) 비트가 된다. 그리고 부호화하는 과정에서 부호화하기 이전에 부호 번호(code number : *N*)라고 하는 색인으로 사상되는데, 이때 부호 번호에 대한 *M*과 *INFO*는 식 (2), (3)와 같이 정의된다.

$$M=ceil(log2(N+1)) \quad (2)$$

$$INFO=N+1-2^M \quad (3)$$

여기서, *ceil*(NUM)는 NUM 보다 작은 가장 근접한 정수 값을 갖는다. 이와 같은 관계식에 의하여 전송된 부호어는 복호화 과정에서는 다음 순서로 복원된다.

i) [1]에 앞서 뒤따르는 선두 '0'의 *M* 비트 수를 읽는다.
ii) *M*-비트 *INFO* 영역을 읽는다.
iii) $N=2^M+INFO-1$ (4)

예를 들어 RUN 개수 값이 4 라고 하면, 이 값 자체가 그대로 부호 번호로 사상되어 *N*=4가 된다. 그러므로 부호화 과정에서 식(2), (3)에 의하여 *M*=2, $INFO=1_{10}=01_2$가 된다. 즉 *[M 0s]=[00]*, *INFO=[01]*이 되므로 최종적으로 할당되는 부호어는 식 (1)에 의해 00101이 된다. 반대로 복호화 과정에서는 i), ii) 절차에 의하여 *M*=2, *INFO* $=01_2=1_{10}$이 되어, 식 (4)에 의해 부호 번호(*N*)는 4로 복원한다. 이는 부호화기나 복호화기에 별도의 VLC 테이블이 없이도 쉽게 부호화, 복호화가 가능하다.

Ⅲ. 제안 알고리즘

H.264 Exp-Golomb 부호화는 양자화된 예측 오차 신호를 제외한 모든 매크로 블록층 구문 요소들에 대하여 독립적으로 적용한다. 이러한 이유로 부호화하는 구문 요소들 사이의 상관관계는 전혀 고려되지 않는다. 그리고 정적 확률 분포 특성을 지닌 고정된 부호어로 부호화하므로, 부호화하는 조건에 따라 변화하는 각 구문 요소들의 확률 분포에 대해서는 대응하지 못한다. 특히 매크로 블록층 구문 요소 중에서도 RUN 구문 요소는 다른 구문 요소와는 달리 연속 길이 부호화를 적용하고 있다. 일반적으로, 발생되는 부호들을 그대로 부호화하는 경우 부호당 1비트 이상 소요된다. 이를 연속 길이로 변환한 후 부호화할 경우 부호당 평균 비트수는 사용된 부호의 평균 비트수를 평균 연속 길이로 나눈 값이 되어 1비트 이하로 줄일 수가 있다[3]. 그러나 일반적으로 움직임이 많고 복잡한 영상에서는 생략 매크로 블록의 발생 빈도수가 매우 적다[4]. 이는 상대적으로 생략 매크로 블록의 평균 연속 길이를 작게 한다. 따라서 RUN 구문 요소에 대한 부호당 평균 비트수를 증가하는 결과를 초래하므로 최적 성능의 압축 효과를 기대하기란 어렵다. 표 1은 이와 같은 특성을 따르는 foreman 동영상이 다른 동영상에 비해 RUN 구문 요소에서 현저히 부호화 효율(Coding Efficiency)이 낮음을 보여준다.

표 1. RUN 구문 요소에 대한 부호화 효율 [H,L_{avg}:비트]

Image sequences	Efficiency(x)=H(x) / Lavg(x)	QP16	QP20	QP24	QP28
Container (QCIF, 100frames, 10fps)	H(RUN)	1.950	2.138	2.484	3.209
	Lavg(RUN)	2.103	2.255	2.582	3.368
	Efficiency(RUN)	**0.927**	**0.948**	**0.962**	**0.953**
Foreman (QCIF, 100frames, 10fps)	H(RUN)	0.293	0.470	0.643	0.870
	Lavg(RUN	1.088	1.159	1.240	1.367
	Efficiency(RUN)	**0.269**	**0.406**	**0.519**	**0.636**
News (QCIF, 100frames, 10fps)	H(RUN)	1.927	2.201	2.519	2.809
	Lavg(RUN)	2.144	2.376	2.678	3.080
	Efficiency(RUN)	**0.899**	**0.926**	**0.941**	**0.912**
Silent (QCIF, 150frames, 15fps)	H(RUN)	1.966	2.182	2.522	2.923
	Lavg(RUN)	2.193	2.431	2.833	3.348
	Efficiency(RUN)	**0.896**	**0.898**	**0.890**	**0.873**

이를 개선하기 위해서 본 논문에서는 부호화하는 매크로 블록이 크게 움직임 보상 매크로 블록과 생략 매크로 블록으로 나뉜다는 사실을 근거로 하였다. 따라서 이와 관련된 구문 요소로서 MB_type와 RUN 구문 요소간의 상관성 여부를 조사하였다. 이를 위해 본 논문에서는 기존의 독립된 두 구문 요소의 엔트로피 합인 H(RUN)+H(MB_type)와 이들 두 구문 요소를 쌍으로 하는 하나의 (RUN, MB_type) 구문 요소의 엔트로피 H((RUN, MB_type))를 실험을 통해 구하였다. 그 결과, 표 2와 같이 H(RUN)+H(MB_type) > H((RUN, MB_type)) 관계식이 성립됨을 확인하였다.

표 2. 기존의 독립된 RUN, MB_type와 (RUN, MB_type) 구문 요소에 대한 엔트로피 비교 [비트]

Image sequences	Entropy H(x)	QP16	QP20	QP24	QP28
Container (QCIF, 100frames, 10fps)	H(RUN)	1.950	2.138	2.484	3.209
	H(MB_type)	2.471	2.341	2.168	1.946
	H(RUN) + H(MB_type)	**4.421**	**4.479**	**4.652**	**5.155**
	H((RUN, MB_type))	**4.249**	**4.256**	**4.396**	**4.826**
Foreman (QCIF, 100frames, 10fps)	H(RUN)	0.293	0.470	0.643	0.870
	H(MB_type)	2.591	2.555	2.381	2.209
	H(RUN) + H(MB_type)	**2.884**	**3.025**	**3.024**	**3.079**
	H((RUN, MB_type))	**2.865**	**3.002**	**3.008**	**3.060**
News (QCIF, 100frames, 10fps)	H(RUN)	1.927	2.201	2.519	2.809
	H(MB_type)	2.372	2.516	2.606	2.683
	H(RUN) + H(MB_type)	**4.299**	**4.717**	**5.125**	**5.492**
	H((RUN, MB_type))	**4.172**	**4.547**	**4.913**	**5.230**
Silentr (QCIF, 150frames, 15fps)	H(RUN)	1.966	2.182	2.522	2.923
	H(MB_type)	2.340	2.427	2.459	2.495
	H(RUN) + H(MB_type)	**4.306**	**4.609**	**4.981**	**5.418**
	H((RUN, MB_type))	**4.184**	**4.476**	**4.814**	**5.186**

이는 RUN과 MB_type 구문 요소 사이의 상관관계가 있음을 간접적으로 보여준다. 그리고 두 구문 요소를 그림 5와 같이 기존의 (a)방식에 비해 하나의 (RUN, MB_type) 구문 요소로 다루는 (b)방식이 보다 압축 성능이 개선될 수 있음을 말해준다.

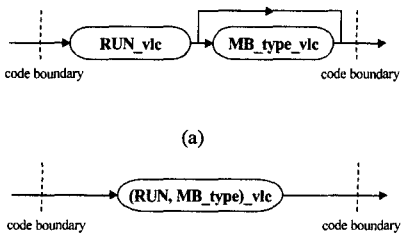

그림5. RUN, MB_type 구문 요소에 대한 기존 방식 (a)과 제안 방식 (b)

한편, H.264에서는 부호화할 화면마다 마지막 매크로 블록이 그림 6과 같이 생략 매크로 블록이면 추가적으로 화면 종료 신호로서 RUN 개수 값(예. 63)을 이용하여 RUN 구문 요소로 Exp-Golomb 부호화하여 복호화기로 전송한다.

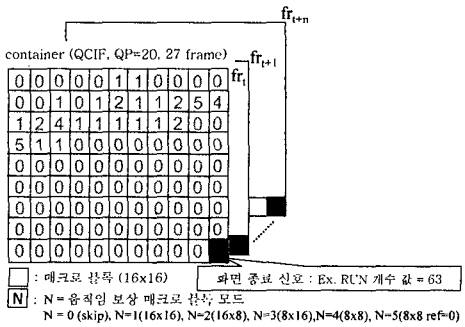

그림6. 부호화할 화면에 대한 화면 종료 신호의 한 예

그러므로 이들 신호에 대한 RUN 개수 값의 발생 확률 분포 특성이 평균 비트 수에 영향을 미친다. 그러나 일반적으로 움직임이 적고 평탄한 영상에서의 화면 종료 신호에 대한 RUN 개수 값의 발생 확률 분포 특성은 그림 7과 같이 매우 불규칙적이다.

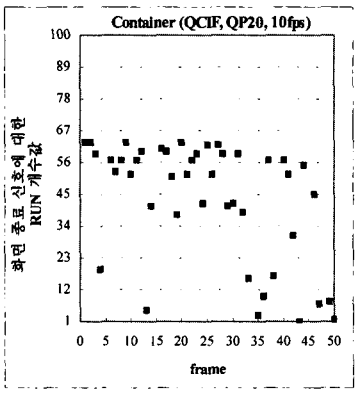

그림7. 화면당 발생하는 화면 종료 신호에 대한 RUN 개수 값의 확률 분포 특성

이는 전반적으로 평균 비트 수를 증가시키는 결과를 초래한다. 그러나 이들 신호를 하나의 EOF (End-Of-Frame) 신호로 다루어, 그 발생 확률에 따라 고정된 부호어 길이로 할당한다. 이렇게 할당된 비트 수가 기존의 평균 비트수보다 작다면 평균 비트 수는 감소하게 될 것이다. 그림 8는 앞서 그림 7의 화면 종료 신호에 대한 평균 비트 수가 약 10.14 비트인 반면에, 이들 신호를 하나의 EOF 부호로 다루었을 때 비트 수는 5 비트로 평균 비트 수가 감소된다.

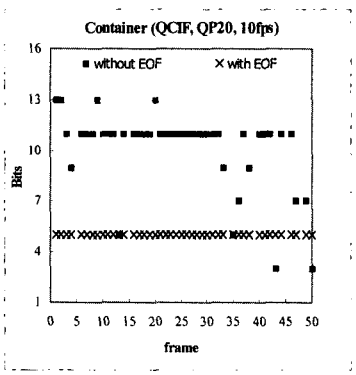

그림8. 화면당 발생하는 화면 종료 신호에 대한 기존 방식과 EOF 신호로 하는 제안 방식에서의 비트 수

그러나 이와 같이 EOF 신호로 다루어 부호화하는 방식이 움직임이 작고 평탄한 영상에서는 효과적인 반면, 움직임이 많고 복잡한 영상에서는 오히려 평균 비트 수의 증가가 예상된다. 그러나 이 경우 생략 매크로 블록의 발생 빈도수는 거의 미비하므로 손실은 거의 없다. 따라서 앞서 언급한 (RUN, MB_type) 구문 요소와 EOF 부호를 접목한, 보다 효과적인 새로운 {(RUN, MB_type)+EOF} 구문 요소를 제안한다. 표 3은 이 제안한 구문 요소에 대한 엔트로피를 조사한 결과로, H(RUN, MB_type) ≥ H({(RUN, MB_type)+EOF})의 관계식이 성립함을 보여준다.

표 3. 제안한 {(RUN, MB_type)+EOF} 구문 요소에 대한 엔트로피 : H({(RUN, MB_type)+EOF}) [비트]

Image sequences	Entropy H(x)	QP16	QP20	QP24	QP28
Container (QCIF, 100frames, 10fps)	H(RUN) + H(MB_type)	4.421	4.479	4.652	5.155
	H((RUN, MB_type))	4.249	4.256	4.396	4.826
	H({(RUN, MB_type) + EOF})	**4.196**	**4.115**	**4.205**	**4.530**
Foreman (QCIF, 100frames, 10fps)	H(RUN) + H(MB_type)	2.884	3.025	3.024	3.079
	H((RUN, MB_type))	2.865	3.002	3.008	3.060
	H({(RUN, MB_type) + EOF})	**2.865**	**3.002**	**3.008**	**3.060**
News (QCIF, 100frames, 10fps)	H(RUN) + H(MB_type)	4.299	4.717	5.125	5.492
	H((RUN, MB_type))	4.172	4.547	4.913	5.230
	H({(RUN, MB_type) + EOF})	**4.133**	**4.476**	**4.790**	**5.066**
Silent (QCIF, 150frames, 15fps)	H(RUN) + H(MB_type)	4.306	4.609	4.981	5.418
	H((RUN, MB_type))	4.184	4.476	4.814	5.186
	H({(RUN, MB_type) + EOF})	**4.146**	**4.412**	**4.708**	**5.006**

결론적으로, 본 논문에서는 H.264에서 기존의 독립적으로 다루어 부호화하던 RUN, MB_type 두 구문 요소를 그림 9와 같이 새로운 하나의 {(RUN, MB_type)+EOF} 구문 요소로 부호화하는 방식을 제안한다.

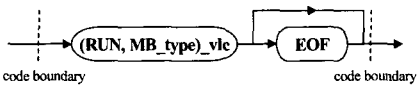

그림9. 새로운 {(RUN, MB_type)+EOF} 구문 요소의 제안 부호화 방식

Ⅳ. 실험 결과 및 고찰

본 논문에서는 H.264 JM2.0 소프트웨어를 사용하여 모의실험을 수행하였다. 실험 환경으로는 동영상에 해상도가 176x144인 QCIF 형식의 container, foreman, news, silent를, 양자화 크기는 16, 20, 24, 28로 선택하였다. 그리고 사용된 총 프레임(frame) 수로는 300 frms, 프레임 율은 container, foreman, news : 10 fps, silent : 15 fps로 하였다. 또한 움직임 추정 및 보상에 사용된 참조 화면 수는 5로 하였다. 이와 같은 실험적 환경은 JVT의 권고 안을 따랐다[5].

그림 10은 동영상 종류나 양자화 크기에 따라 다소 성능 차이는 있지만, 두 RUN, MB_type 구문 요소에 대하여 독립적으로 다루던 기존 방식에 비해 본 논문에서 제안한 {(RUN, MB type)+EOF} 구문 요소로 다루어 부호화하는 방식이 평균 압축률이 크게 개선되는 것을 보여준다.

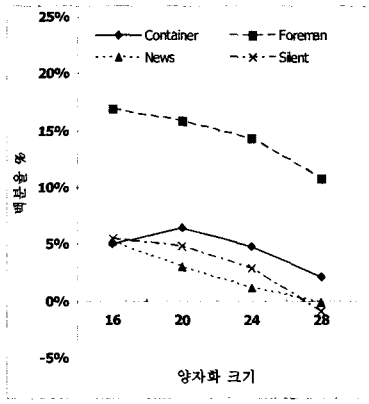

그림10. RUN, MB_type 구문 요소를 독립적으로 부호화하던 기존 방식 대비 제안한 구문 요소로 부호화한 제안 방식에 대한 두 구문 요소에 있어서의 평균 비트 감소율

특히 움직임이 많고 복잡한 영상 또는 저 압축률에서는 생략 매크로 블록의 발생 확률이 낮다는 사실을 감안한다면, 연속 길이 부호화를 적용한 RUN 구문 요소에 대해서는 효율적인 압축 성능을 기대하기란 어렵다. 그러나 본 논문에서 제안한 {(RUN, MB_type)+EOF} 구문 요소는 RUN과 MB_type 구문 요소 사이의 상관관계를 이용하였다. 이는 움직임이 많고 복잡한 foreman 과 같은 동영상에서 평균 15 %로 크게 압축률이 향상되는 효과를 얻었다. 이에 반해 움직임이 적고 평탄한 영상 또는 고 압축률에서는 기존 방식에 비해 다소 압축 성능이 떨어진다. 그러나 본 논문에서 제안한 {(RUN, MB_type)+EOF} 구문 요소의 발생 확률 분포 특성을 고려한다면 보다 향상된 압축 성능 개선 효과가 기대된다. 그림 12는 전체적인 비트 감소율로, 화질의 열화가 전혀 없이 최대 1.6 %의 압축 성능 향상이 있음을 보여준다.

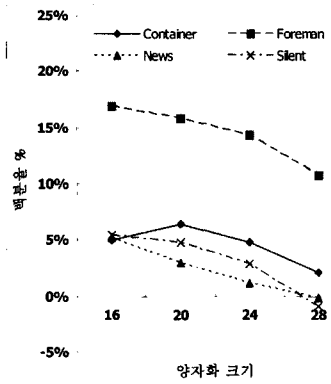

그림12. RUN, MB_type 구문 요소를 독립적으로 부호화하던 기존 방식 대비 제안한 구문 요소로 부호화한 제안 방식의 전체 평균 비트 감소율

V. 결 론

본 논문에서는 H.264 저 전송률 동영상 압축 방식에서 기존의 독립된 RUN, MB_type 두 구문 요소 사이의 상관관계를 이용하여 하나로 통합한 (RUN, MB_type) 구문 요소를 제시하였다. 또한 보다 개선된 압축 성능을 위하여 화면 종료 신호로서 EOF 부호를 접목한 효과적인 새로운 {(RUN, MB_type)+EOF} 구문 요소를 제안하였다. 그리고 본 논문에서 제안한 {(RUN, MB_type)+EOF} 구문 요소를 실험적 사실을 통한 엔트로피 관점에서 검증하였다. 결론적으로, 두 구문 요소 RUN, MB_type를 기존의 독립적으로 부호화하던 방식에 비해 제안한 {(RUN, MB_type)+EOF} 구문 요소로 부호화하는 방식이 다양한 동영상과 압축률 조건에서 효과적이라는 사실을 보여준다.

모의실험에서는 H.264 Exp-Golomb 부호화한 결과, 부호화 조건에 따라 다소 성능 차이는 있었지만 기존보다 화질의 열화가 전혀 없이 최대 15%의 압축률이 향상됨을 보였다. 그러나 보다 개선된 압축 성능 향상을 위해서는 본 논문에서 제안한 {(RUN, MB_type)+EOF} 구문 요소의 발생 확률 분포를 따르는 효율적인 부호화 방식에 대한 연구가 지속적으로 필요하다고 생각된다.

참 고 문 헌

[1] Joint Video Team(JVT) of ISO/IEC MPEG and ITU-T VCEG, "Editor's Proposed Draft Text Modifications for Joint Video Specifica -tion (ITU-T Rec. H.264| ISO/IEC 14496-10 AVC), Geneva modification, draft 7", Oct. 2002.

[2] P. Topiwala, G. Sullivan, A. John, F.Kosse -ntini, "Overview and Performance Evalution of the ITU-T Draft H.26L Video Coding Standard," *Proc. SPIE, Appl. of Digital Ima ge Processing*, August 2001.

[3] S.W. Golomb, "Run-Length Coding," *IEEE Trans. on Inf. Theo*, vol 1T-12, pp.399-401, July 1966.

[4] W. Choi and B. Jeon, "Universal Variable Length Coding with Adaptive Codeword Ma pping using Fixed Re-Association Table," *VLVB 2001*, Oct. 2001.

[5] ITU-T, VCEG-N81, "Recommended Simula tion Common Conditions for H.26L Coding Efficiency Experiments on Low-Resolution Progressive-Scan Source Material," Sept. 20 01.

[6] Marpe, D. Schwarz H. Blattermann, G. Hei -sing, G. Wiegand, T, "Context-based adapti ve binary arithmetic coding in JVT/H.26L," *Image Processing*. 2002. Proceedings. *2002 International Conference on*, Volume 2, 22-25 Sept. 2002.

백 성 학(Seong-Hak Baek) 정회원
한국통신학회논문지 Vol. 26, No. 12A 참고

문 용 호(Yong-Ho Moon) 정회원
현재: 부산외국어대학교 디지털정보공학부 전임강사
한국통신학회논문지 Vol. 28, No.5C 참고

김 재 호(Jae-Ho Kim) 정회원
한국통신학회논문지 Vol. 26, No. 11A 참고