2014년 4월 전자공학회 논문지 제 51권 제 4 호
Journal of The Institute of Electronics and Information Engineers Vol. 51, NO. 4, April 2014 http://dx.doi.org/10.5573/ieie.2014.51.4.160

논문 2014-51-4-17

와이너 지브 비디오 압축에서의 비트 플레인 상관관계를 이용한 고속 LDPC 복호 방법

(Fast LDPC Decoding using Bit Plane Correlation in Wyner-Ziv Video Coding)

오 양 근*, 심 혁 재*, 전 병 우**

(Ryanggeun Oh, Hiuk Jae Shim, and Byeungwoo Jeon©)

요 약

와이너 지브(Wyner-Ziv) 비디오 압축은 리소스 제한적인 부호화기를 사용하는 다양한 어플리케이션에 있어 상당히 유용하지만, 와이너 지브 복호화기는 상당량의 복잡도를 요구하는 문제점을 가지고 있다. 이러한 복잡도의 주요 원인으로는 LDPC 채널 복호 과정을 꼽을 수 있는데, 이는 점진적으로 요구되는 패리티량에 따라 매번 반복적인 채널 복호과정을 거치기 때문이라 할 수 있다. 본 논문에서는 이러한 복잡도의 문제를 해결하기 위해 비트 플레인을 두 개의 그룹으로 나누고, 각 그룹의 패리티 요구량의 예측치를 비트 플레인 간의 상관관계에 근거하여 계산한다. 따라서 제안된 와이너 지브 복호화기는 예측된 최소 패리티 예측량에 따른 패리티를 전송 받은 이후 반복적인 복호 과정을 거치게 된다. 본 제안 방법을 적용할 경우 LDPC 복호 과정의 71% 정도의 시간을 절약할 수 있다.

Abstract

Although Wyner-Ziv (WZ) video coding proves useful for applications employing encoders having restricted computing resources, the WZ decoder has a problem of excessive decoding complexity. It is mainly due to its iterative LDPC channel decoding process which repeatedly requests incremental parity data after iterative channel decoding of parity data received at each request. In order to solve the complexity problem, we divide bit planes into two groups and estimate the minimum required number of parity requests separately for the two groups of bit planes using bit plane correlation. The WZ decoder executes the iterative decoding process only after receiving parity data corresponding to the estimated minimum number of parity requests. The proposed method saves about 71% of the computing time in the LDPC decoding process.

Keywords : DVC, Wyner-Ziv, LDPC, Fast decoding, HDA method

* 학생회원, ** 정회원, 성균관대학교 정보통신대학
(College of Information and Communication Engineering, Sungkyunkwan University)
© Corresponding Author(E-mail: bjeon@skku.edu)
※ This work was supported by the National Research Foundation of Korea(NRF) grant funded by the Korea government(MSIP) (No. 2011-001-7578).
접수일자: 2014년2월18일, 수정일자: 2014년3월9일, 수정완료: 2014년3월28일

Ⅰ. INTRODUCTION

In conventional video coding, such as H.26X, an encoder handles heavy processing tasks such as motion estimation (ME) and motion compensation (MC). The complexity of the encoder is much higher than that of the decoder. As a result, the

conventional coding structure is suitable for broadcasting environment under which having substantial computing resource makes sense to be at encoder side. However, under recent broadcasting-communication convergence environment, many new distributed applications such as wireless video cameras in mobile phones, sensor networks, and surveillance systems have appeared. These applications typically have restricted computing resources, therefore, require an encoder with lower complexity.

Distributed video coding (DVC) is a technique developed to meet the requirements of those applications. It is based theoretically on the Slepian-Wolf theorem[1] and the Wyner-Ziv theorem[2] in the 1970s. Slepian and Wolf mathematically proved that, although the prediction information (Y) is given only to a decoder, the minimum information for lossless reconstruction of the original information (X) can be the conditional entropy H(X|Y). The Wyner and Ziv further showed that it is also possible to obtain the same rate-distortion (R-D) performance as that by the predictive encoding in lossy compression. It suggests that even though sources are encoded independently without using joint signal processing methods such as ME and MC, it is possible to attain R-D performance comparable to conventional video coding if the signal is decoded using the correlation of the sources. This means that a simpler encoder can work by performing joint signal processing at the decoder.

Wyner-Ziv (WZ) video coding structure, proposed by Aaron et al.[3], divides input pictures into key and WZ frames. These frames are independently encoded respectively by conventional H.26X intra coding and a channel code, such as Turbo code[4, 23] or low-density parity check (LDPC) code[5]. In the decoding process, the WZ decoder reconstructs two key frames (t-1, t+1) from the transmitted two encoded key frames and then generates side information (SI) from the reconstructed two key frames using motion compensated temporal interpolation (MCTI)[6~7, 22]. SI is considered as a reconstructed WZ frame that includes correlation noise, which is the difference between the SI and the original WZ frame (t). The WZ decoder estimates the noise using a correlation noise model[8~9] and eliminates it in the channel decoding process. Thus, the performance of WZ coding is directly related to the performance of the channel code.

Among existing channel codes, Turbo and LDPC codes provide powerful error-correction capability, which are close to the Shannon limit. These channel codes are based on a stochastically iterative decoding process with soft decisions. This iterative decoding process is quite time consuming, increasing the complexity of a WZ decoder that adopts LDPC or Turbo codes. Currently, the low-density parity check accumulate (LDPCA) code[10] is used in Wyner-Ziv (WZ) video coding instead of LDPC for rate adaptation. A LDPCA decoder assumes about 70% of the total WZ decoder complexity; thus, its complexity reduction that does not decrease R-D performance is a key in solving the complexity problem of WZ decoder.

Several methods have been proposed to reduce the channel decoding complexity. In 2005, [11] proposed using LDPC decoding outputs under a Gaussian distribution for a finite block message. This method determines if a block message is decodable, and reduces unnecessary iterations. However, it does not consider the random noise that occurs in real video data, so its direct application to real video sequences can cause a problem such as degradation of R-D performance. In 2007, [12] proposed a method which reduces Turbo decoding complexity by estimating a decodable rate at an encoder, however, its encoder complexity increases in return. It is in contrast to the concept of DVC, which simplifies the encoder, and it also decreases R-D performance considerably. In 2008, [13] proposed controlling Turbo decoder complexity both at encoder and decoder. Here, the

encoder estimates the rate from three previously decoded WZ frames and transmits the estimated amount of parity data. This method reduces the Turbo decoder complexity but at considerable loss of R-D performance. In 2009, [14] reduced the LDPC decoder complexity at the decoder.

In this paper, we first introduce the latest fast LDPC decoding method[14] and explain a hard-decision aided (HDA) method[15, 24], and then propose a faster LDPC decoding method using bit plane correlation. This paper is organized as follows: Section Ⅱ explains the LDPC channel code; Section Ⅲ, the major contribution of this paper, describes the proposed fast LDPC decoding method; Section Ⅳ gives simulation results; finally, Section Ⅴ presents conclusions and comments on future work.

Ⅱ. Low-Density Parity Check Encoding and Decoding

1. Encoder

A WZ encoder divides a WZ frame into 4x4 blocks, and each block is transformed to the frequency domain by the 4x4 (DCT). The same frequency coefficients in a frame are grouped together and quantized by a quantization matrix (QM)[16]. From MSB to LSB, each bit plane is given to the LDPC encoder in the frequency band order (zigzag scan order). LDPC is characterized by a parity check matrix H with dimension mxn. It performs $m=n-K$ parity checks, where K is the amount of input data and n is total amount of data (where, $n-K$ is redundant). Multiplying codeword x, Hx is supposed to be 0[17]. The syndrome is calculated as $s=Hx$.

Each bit plane (x) which is input to the LDPC encoder is regarded as message node bit. Given H, to encode x, the LDPC encoder calculates the syndrome s. Here, the LDPCA code is used instead of LDPC code to transmit the syndrome at adaptive rate. The LDPCA encoder accumulates syndrome bits by modulo 2 and produces the accumulated syndrome a, which is called parity data. Quarter common intermediate format (QCIF) sized sequences, for instance, generate 1584 parity bits which are stored in an encoder buffer and transmitted adaptively, according to requests from the decoder[18].

2. Decoder

WZ decoder reconstructs key frames with the H.26X intra decoder, and then generates SI based on the reconstructed key frames using MCTI[6~7]. SI is regarded as a noisy version of the WZ frame. Since there is no real WZ frame information at the decoder, the difference between two key frames is assumed as the correlation noise[9]. Laplacian distribution is widely used to model the correlation noise in the WZ video coding, which is represented as equation (1):

$$P(u)=\frac{\alpha}{2}e^{-\alpha|u|} \tag{1}$$

where u is a random variable representing the correlation noise and the Laplacian distribution parameter α is defined by,

$$\alpha=\sqrt{\frac{2}{\sigma^2}} \tag{2}$$

where σ^2 is the variance of the noise which is difference between two neighboring key frames[6]. The LDPC decoder computes the intrinsic log-likelihood ratio (LLR) using the SI and Laplacian distribution (its parameter is α) as shown in equation (3)[6]. The LLR represents log scale probability ratio of whether reconstructed bit is likely to be a zero or one.

$$LLR=\log\left[\frac{P(X=0|Y)}{P(X=1|Y)}\right] \tag{3}$$

In equation (3), P is the Laplacian probability density function and P(X=0|Y) is the conditional probability that the reconstructed bit X=0 when SI (Y) is given. The LDPC decoder starts decoding with

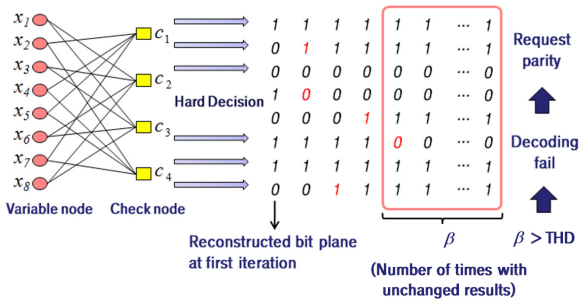

그림 1. HDA 개념도
Fig. 1. The HDA concept.

the intrinsic LLR and transmitted parity. The intrinsic LLR becomes the input of variable nodes that represent the codeword bits. In the first half of the decoding process, a product algorithm, which is a parity check equation, is executed at check nodes using transmitted messages from variable nodes connected by the parity check matrix H. In the second half, the sum algorithm is executed at the variable nodes using the transmitted messages from connected check nodes and the LLR output is calculated[19]; hence the LDPC decoding process is called a sum-product algorithm, message passing, or belief propagation algorithm. The LLR output is updated according to the results of the sum-product algorithm; messages pass between variable nodes and check nodes. For further detail, refer to[8, 10, 21].

Reconstructed bits are generated by hard decisions of the LLR output, and a convergence test is performed on the reconstructed bits[18]. In the convergence test, the LDPC decoder calculates the bit-error rate (BER). If the calculated BER is less than a target one (e.g., 10^{-4}), then the decoder checks the reconstructed bit plane with cyclic redundancy check (CRC) bits to confirm the error-correction results. If the CRC matches, the decoding of the current bit plane is completed; however, if the calculated BER is higher than the target one, the LDPC decoder again carries out the iterative sum-product process with the previous LLR output [10]. This iterative decoding is performed up to a predefined maximum number of times. In this paper, we use a maximum of 50 iterations.

If the calculated BER never falls below the target BER even after the maximum number of iterations, the LDPC decoder regards this as a decoding failure for the parity data already received, and requests more parity bits from the encoder. The encoder transmits a small number of additional parity bits which are stored in a buffer, and the decoder uses them to execute iterative decoding again.

The amount of transmitted parity data responding to one parity request is denoted by M_0 and it is calculated as in equation (4).

$$M_0 = \frac{K}{R_{Max}} \tag{4}$$

where K is the length of input data x (also, total parity data length), and R_{Max} is the maximum number of parity transmission allowed. For a QCIF frame, when using the LDPCA code of length 1584, K is 1584 and R_{Max} is 66. Therefore M_0 is 24 bits (24=1584/66). This paper explains the fast LDPC decoding method using the LDPCA code of length 1584.

Iterative decoding is quite time consuming. If the decoder requests parity from the encoder R_{Max} (= 66) times, the maximum number of iterations can be 3,300 (66 requests ×50 iterations). The LDPC decoder carries out iterative decoding repeatedly until decoding succeeds; thus, the complexity of the WZ decoder increases. In case of QM 8[16], the LDPC decoder comprises more than 70% of the total complexity of the WZ decoder.

Ⅲ. Fast LDPC Decoding

Several research groups have proposed methods of reducing LDPC decoder complexity for Wyner-Ziv video coding. Among these, Ascenso et al.[14] proposes a state-of-the-art method for reducing channel decoding complexity. It calculates an average of the absolute differences of LLR values between

iterations and the number of satisfied parity check equations at the end of the iterations. They first check the average of LLR differences and the number of unsatisfied parity check equations during each iteration process. If the average of LLR differences is smaller than a predefined threshold λ, then, one early stopping parameter L is increased by one; if the number of unsatisfied parity check equations is equal to that of the previous iterations, the other early stopping parameter C is increased. Finally, if either L or C is larger than a predefined parameter d, the source is classified as not decodable, and the iterative algorithm is stopped. The parameters λ and d were experimentally selected as 0.5 and 6, respectively. Note that this method depends on adjusting the thresholds λ and d, and the performance varies according to the configured λ and d. In general, the faster LDPC decoding it becomes, the worse R-D performance it obtains. It is hard to find balanced threshold parameters that simultaneously reduce the computing time and R-D performance drop. Therefore, a new requirement for a fast method should be to save significant computing time with only a negligible R-D performance degradation.

1. HDA Method

There have been many methods proposed to reduce the complexity of the channel decoder. Among them, the hard-decision aided (HDA) method[15] reduces the complexity of iterative decoding of Turbo code. Like the Turbo code, LDPC code also requires an iterative decoding based on the stochastic sum-product calculation; thus, it is possible to apply the HDA method for the LDPC code as well.

The HDA method is based on the LLR output, which is the iterative decoding output using intrinsic LLR; the LLR output is updated after each iteration. Figure 1 illustrates the concept of HDA, where the left side represents LDPC iterative decoding and the right represents the HDA early stopping criterion. On the left side, the x and c nodes are the variable and

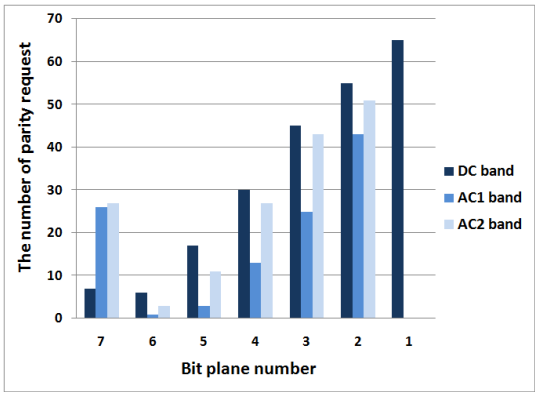

그림 2. 패리티 요구량 (Foreman, 1번째 프레임)
Fig. 2. The number of parity requests (Foreman, the first frame), DC (7: MSB, 1: LSB of magnitude), AC1 & AC2 (7: sign bit; 6: MSB-1, 2: LSB of magnitude).

check nodes, respectively. The intrinsic LLR (only the first iteration) or LLR output becomes the input of the variable nodes. In check nodes, parity check equations are calculated with parity data and messages transmitted from variable nodes. In the variable nodes, the LLR output is calculated using the messages transmitted from the check nodes. The calculated LLR output becomes the reconstructed bit plane through the hard decision process and BER is calculated from it. If the calculated BER is smaller than a target BER, then the LDPC iterative decoding process is terminated. If the calculated BER is higher than the target BER and the reconstructed bit plane does not change from the one obtained at the previous iteration, then the number of iterations having successively unchanged decoding results, β, is increased by one. If β is larger than a predefined parameter THD, which is called a HDA condition, the decoder determines that the current parity data is insufficient, so it stops the algorithm and requests more parity data from the encoder.

Using HDA, the LDPC decoder does not need to be executed up to the maximum number of iterations, thus making it very effective in terms of complexity reduction. However, it is not sufficiently effective since it concerns only about stopping the iterations before reaching the maximum number of iterations,

therefore quite a few number of iterative process must be performed as parity is kept being received over multiple parity requests until reaching any meaningful stage in which the checking HDA condition makes sense. Therefore, there must be some limits in complexity reduction if only the HDA method is used. It should be very desirable to estimate a meaningful minimum number of parity requests in order to skip unnecessary iterative decoding until receiving the estimated necessary amount of parity.

We define R_{Min}, an estimated minimum number of parity requests. The LDPC decoder is to receive the parity data corresponding to R_{Min} without actual performing iterative decoding. Now we like to find a way to estimate R_{Min}.

2. Estimation of Minimum Number of Requests using Bit Plane-wise Correlation

HDA reduces the number of LDPC decoding iterations required for successful decoding, thus reducing the decoder complexity; however, the LDPC decoder still accounts for large amount of the complexity of the WZ decoder. An additional algorithm is required to reduce the complexity with a negligible loss of R-D performance.

Figure 2 shows the number of parity requests until decoding success in the DC, AC1, and AC2 bands (in zigzag scan order) of the Foreman sequence (the first frame). Quantization is done to DCT coefficients using QM8[16], therefore, DC symbol is assigned 7 bits, and each AC1 and AC2 band symbol is assigned 6 bits. Decoding starts from MSB (i=7) to LSB (i=1). Quite expectedly, the number of parity requests until decoding success generally increases as the bit plane comes closer to the LSB side because of increased correlation noise in lower bit planes. In Figure 2, we can observe that for AC, the number of parity requests in MSB is larger than MSB-1. It happens because LDPC decoding is performed using bit plane-based dependency and previously decoded results (i.e. upper bit planes). Therefore, the benefits from the bit-plane dependency and previously decoded results can decrease the number of parity requests in MSB-1, but the MSB bitplane cannot utilize this information. Moreover, there is one difference between the DC and the AC bands: while the DC band has only a magnitude value, AC bands have a sign (MSB) and magnitude (other bit planes) values. This is why the numbers of parity requests in MSB and MSB-1 in the AC band shows much larger differences than the DC band.

The minimum number of parity requests is estimated using a bit plane-wise correlation inside a symbol except for MSB and MSB-1. When the LDPC decoder starts decoding the i-th bit plane, it is initially informed of the number of parity requests required for successful decoding of its previous bit plane (that is, (i+1)-th) unless it is the first bit plane which has no upper bit plane. A rough estimate of the minimum number of parity requests, R_B, is set as the number of parity requests actually happened for successful decoding of its upper bit plane. Suppose the number of parity requests for the (i+1)-th bit plane was 10, then, while the conventional method has to perform a maximum of 450 (9 x 50 times) iterations for the decoding i-th bit plane, the proposed estimated minimum number of requests makes it possible to skip those 450 iterations and start decoding from the tenth parity data (R_B =9) directly.

It is impossible, as mentioned above, to apply this fast method to MSB and MSB-1. Therefore, the method above is more effective for a high bit rate case (e.g. QM8) than for a low one (e.g. QM1). The ratio of 2 (MSB, MSB-1) over the total number of bit planes is relatively larger in low bit rate than in high bit rate. Moreover, the LDPC decoder still requires a fair amount of complexity for both high and low bit rates. This indicates that an additional fast method for MSB and MSB-1 is desired to reduce the complexity.

3. Estimation of Minimum Number of Requests using Correlation Noise Model

A WZ decoder creates SI based on the reconstructed key frames using MCTI. Reconstructed SI is regarded as a WZ frame that contains correlation noise, modeled as Laplacian distribution [8~9]. An LDPC decoder computes the intrinsic LLR using the SI and the Laplacian distribution parameter α as in eq.(3). It illustrates the conditional probabilities that a reconstructed bit is zero or one, so it is possible to model their relationship as a binary symmetric channel (BSC)[20]. If the conditional probability of X=0, given SI (Y), is assumed as P_{Ck} (crossover probability), we can rewrite equation (3) as:

$$LLR_{intrinsic,k} = \log\left[\frac{P_k(X_k=0|Y)}{P_k(X_k=1|Y)}\right] = \log\frac{Pc_k}{1-Pc_k} \tag{5}$$

where k is an index indicating bit location (e.g., 1~1584) in the input data x. Since the binary entropy function is symmetric with respect to p=0.5, the entropies of P_{Ck} and $1-P_{Ck}$ are equal. The entropy of the k-th bit is represented as[12]:

$$H_k = -Pc_k \times \log_2 Pc_k - (1-Pc_k)\times\log_2(1-Pc_k) \tag{6}$$

H_k is an information quantity for decoding success of the k-th bit; the total information quantity for the bit plane is then represented as eq.(7):

$$\sum_{k=1}^{K} H_k \tag{7}$$

The total information quantity in eq.(7) represents how many parity bits are needed at least for decoding success of the bit plane. With a fixed amount of parity data by one request, M_0 in eq.(4), it is possible to derive a minimum number of parity requests for the current bit plane. The minimum number of parity requests using the correlation noise mode, R_N, is now estimated by:

$$R_N \cong \frac{\sum_{k=1}^{K} H_k}{M_0} = \frac{\sum_{k=1}^{K} H_k}{K} \times R_{Max} \tag{8}$$

In eq.(8), K is the length of input data x to the LDPC encoder, and R_{Max} is the maximum number of parity requests. An LDPC decoder initially receives the estimated parity data before performing any iterative decoding. In QCIF resolution, if the average H_k of a bit plane is 1, then, R_N is 66.

Figure 3 illustrates a distribution of the estimated minimum number of parity requests using the correlation noise model, R_N, versus the number of parity requests resulting from actual conventional decoding method[21] (hereafter, called the real number of parity requests) using Foreman, Coastguard, Hall monitor, and Stefan sequences (all frames, all bit planes). The horz.-axis is R_N and the vert.-axis is the real number of parity requests. Figure 3 shows that the shape of the distribution is broad and R_N is not accurate. Based on the y=x line, the upper triangle area of the line illustrates the cases of underestimated parity requests, R_N. In this area, there

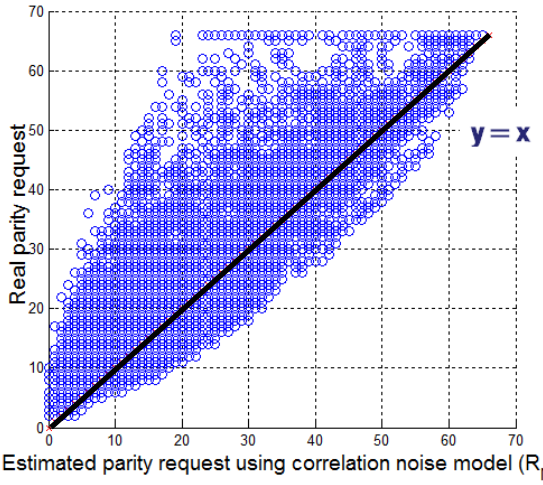

그림 3. 실제 패리티 요구량과 잡음 상관도를 이용한 패리티 예측량

Fig. 3. Estimated number of parity requests using the correlation noise model, R_N vs. the real number of parity requests (four test sequences, all bit planes).

is no degradation in R-D performance since R_N is always smaller than the real number of requests. Moving further away in this area from the y=x line, however, means decrement in complexity reduction performance. The lower triangle area below the y=x line on the other hand illustrates overestimation of parity requests R_N, which has more complexity reduction but always degrades R-D performance. Considering the complexity reduction and R-D performance together, the upper area near y=x is a better target; and the area below the line is the least desirable case.

Figure 3 also shows that, if a decoder starts decoding with the initially transmitted parity data of amount R_N times M_0, its R-D performance may decrease if R_N is overestimated. This means that the problem of R-D performance degradation is not solved yet; thus, it is hard to directly apply the minimum request estimation method using the correlation noise to reduce LDPC decoder complexity.

4. Proposed Fast LDPC Decoding Method

The proposed fast LDPC decoding method basically utilizes the HDA method. As mentioned before, the HDA method is able to minimize the number of iterations corresponding to currently requested (or received) parity data, however, the HDA method itself does not estimate the minimum number of parity requests, R_{Min}. Therefore, in the previous clauses, we have suggested two minimum request estimation methods of R_{Min}. The first one is estimation by the bit plane-wise correlation where corresponding estimated number of parity request is denoted by R_B in former clause III-B. However, the bit plane-wise correlation and its estimation result R_B are only restrictively applicable: that is, it can be applied only to MSB-2 to LSB. The second estimation is by a correlation noise model where its corresponding estimation is denoted by R_N. Different from R_B, the estimated minimum request R_N does not have any restriction to apply to any bit plane, however, its problem is its aforementioned accuracy. Therefore, here we explain how to overcome the problems of restricted application of R_B and inaccuracy of R_N. Consequently, we propose a fast LDPC decoding method based on the HDA method utilizing the minimum request estimation methods using both bit plane-wise correlation and the correlation noise model. In order to reduce the LDPC decoder complexity while preserving R-D performance, the proposed method uses a request estimation method using the bit plane-wise correlation and the correlation noise model that depends on the index of a bit plane. Bit planes are divided into two groups: the M group (MSB and MSB-1) and the L group (the other bit planes).

In the M group, only a fast method using entropy is applicable. Figure 4 shows R_N versus the real number of parity requests in MSB and MSB-1. If R_N is used to estimate the minimum number of parity requests, R_{Min}, there may be a large loss in R-D performance. Since there are few overestimated points below the y=(1/2)x line as seen in Figure 4, to avoid degradation of R-D performance, half of R_N is set to

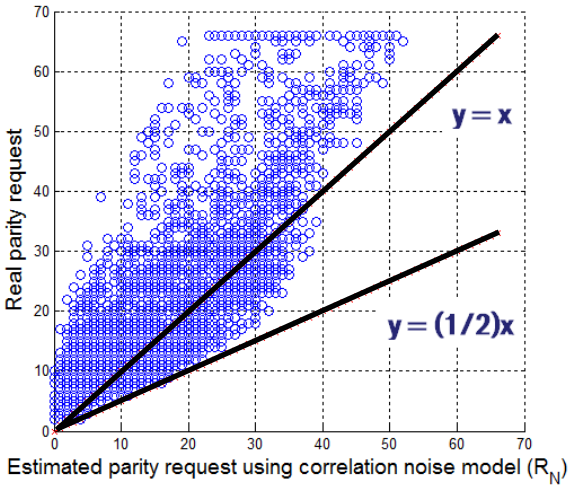

그림 4. 실제 패리티 요구량과 잡음 상관도를 이용한 패리티 예측량 R_N (MSB 와 MSB-1)

Fig. 4. Estimated number of parity requests using the correlation noise model, R_N vs. the real number of parity requests (four test sequences, MSB and MSB-1).

R_{Min}, as shown in eq.(9).

$$R_{Min} = \frac{R_N}{2} \ (for\ M\ group : MSB, MSB-1) \tag{9}$$

In the case of the L group, it is possible to utilize both request estimation methods using the correlation noise model and the bit plane-wise correlation. If the estimated minimum number of parity requests using the bit plane-wise correlation, R_B, is larger than R_N, R_B becomes R_{Min}. Figure 5 shows R_B versus the real number of parity requests, and it shows few points are overestimated. When $R_B > R_N$, R_N is not desirable as an (initial) estimated minimum requests because it is more underestimated than R_B and, therefore, is inappropriate for complexity reduction. On the other hand, if R_N is larger than or equal to R_B, an average of R_N and R_B is set to R_{Min}. Figure 6(a) shows R_N versus the real number of parity requests; if R_N is regarded as R_{Min}, R-D performance degrades significantly. Figure 6(b) shows an average of R_N and R_B versus the real number of parity requests – there are few overestimated points. When $R_B \leq R_N$, an average of R_N and R_B is more efficient than using only R_B, because R_B has more underestimated points compared with the average of R_N and R_B. Eq.(10) estimates the minimum number of parity requests R_{Min} for the L group.

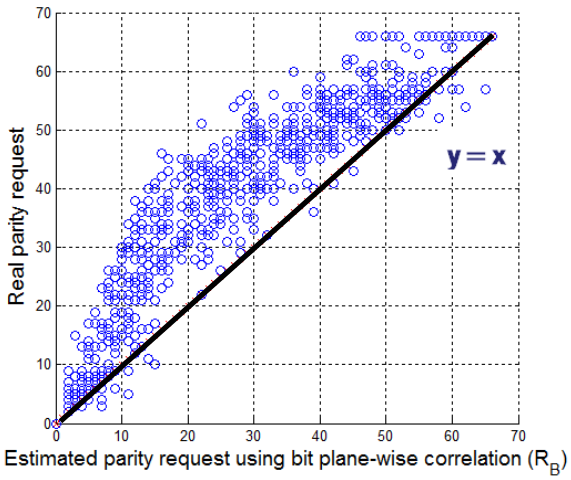

그림 5. 실제 패리티 요구량과 비트 플레인 상관관계를 이용한 패리티 예측량 R_B (MSB-2 에서 LSB)

Fig. 5. Estimated number of parity requests using the bit plane-wise correlation, R_B vs. the real number of parity requests when $R_B > R_N$ (four test sequences, from MSB-2 to LSB).

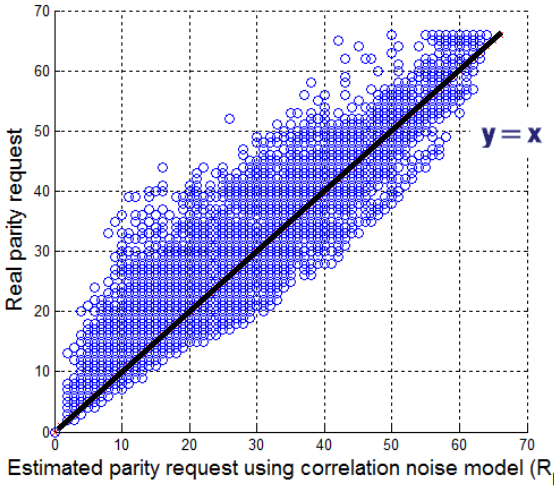

그림 6(a). $R_B \leq R_N$ 인 경우의 실제 패리티 요구량과 잡음 상관도를 이용한 패리티 예측량 (MSB-2 에서 LSB)

Fig. 6(a). Estimated number of parity requests using the correlation noise model, R_N vs. the real number of parity requests when $R_B \leq R_N$ (four test sequences, from MSB-2 to LSB).

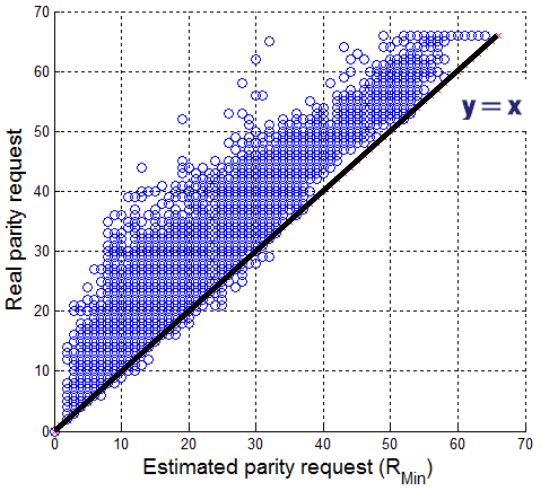

그림 6(b). $R_B \leq R_N$ 인 경우의 실제 패리티 요구량과 제안된 최소 예측량 (MSB-2 에서 LSB)

Fig. 6(b). Estimated number of parity requests R_{Min} vs. the real number of parity requests when $R_B \leq R_N$ (four test sequences, from MSB-2 to LSB).

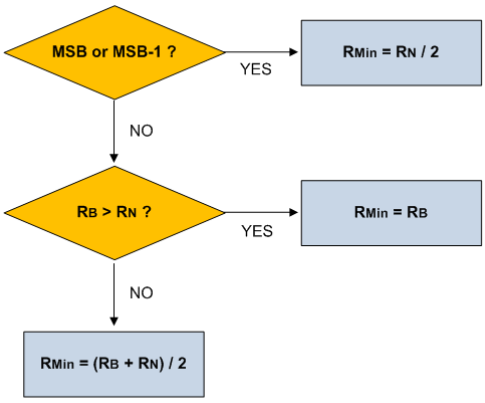

그림 7. 제안 방법의 순서도
Fig. 7. Flow chart of the proposed method.

$$R_{Min} = \begin{cases} R_B & (if\ in\ R_B > R_N,\ L\ group) \\ \dfrac{R_N + R_B}{2} & (if\ in\ R_B \le R_N,\ L\ group) \end{cases} \quad (10)$$

Figure 7 shows a flow chart for calculating R_{Min} depending on the bit plane group. The LDPC decoder is initially informed of the estimated minimum number of parity requests, R_{Min} and starts iterative decoding using the HDA method only after its corresponding amount of initial parity data to R_{Min} is received.

Ⅳ. Experiment and Discussion

1. Experimental Conditions

In this section, we evaluate the complexity reduction (LDPC and WZ decoder) and R-D performance of the latest fast LDPC decoding method [14] and the proposed method. The performance is compared using four different types of sequences (Foreman, Coastguard, Hall monitor, and Stefan). The size of the sequences is QCIF and the frame rate is 15 Hz. The Foreman, Coastguard and Stefan sequences have 150 frames each; Hall monitor has 165 frames. Key frames are encoded with a H.264/AVC intra frame coder; WZ frames are encoded using a LDPCA code[10] with QM 1, 5, 7, 8 [16]. The Foreman and Stefan sequences use QP of 40, 34, 29, 25 corresponding to QM; Coastguard uses QP of 38, 33, 30, 26, and Hall monitor uses QP of 37, 33, 29, 24[21]. The proposed method selects predefined THDs of 6 and 8 for HDA. To compare the proposed method with the latest fast LDPC method[14], the parameters of the latest fast method are varied and performance is compared only for the WZ frames.

표 1. 실험 결과
Table 1. Experimental Results.

Foreman

	Prev	Prev*	HDA6	Prop6	HDA8	Prop8
LDPC TS (%)	68.59	54.50	56.27	65.66	52.88	64.02
WZ TS (%)	42.46	34.03	35.25	41.22	33.16	40.22
BDPSNR [dB]	-0.08	-0.01	-0.004	-0.005	-0.002	-0.003

Coastguard

	Prev	Prev*	HDA6	Prop6	HDA8	Prop8
LDPC TS (%)	73.55	60.52	63.15	72.68	59.52	70.90
WZ TS (%)	37.94	31.56	32.91	38.00	31.11	37.17
BDPSNR [dB]	-0.117	-0.016	-0.009	-0.012	-0.005	-0.008

Stefan

	Prev	Prev*	HDA6	Prop6	HDA8	Prop8
LDPC TS (%)	72.92	59.35	61.10	70.55	57.46	68.95
WZ TS (%)	48.34	39.50	40.75	47.26	38.33	46.24
BDPSNR [dB]	-0.103	-0.013	-0.004	-0.009	-0.002	-0.007

Hall monitor

	Prev	Prev*	HDA6	Prop6	HDA8	Prop8
LDPC TS (%)	75.66	63.31	67.91	76.95	63.84	75.20
WZ TS (%)	28.82	24.33	26.39	30.15	24.78	29.62
BDPSNR [dB]	-0.182	-0.028	-0.016	-0.021	-0.008	-0.014

2. Experimental Results

Table I shows average performance comparison in terms of decoding time saving and PSNR of the latest fast method[14] and the proposed method with the conventional WZ decoding without any fast algorithm as an anchor using QM 1, 5, 7, 8. In Table I, Prev represents the latest fast method[14] with parameters selected as in [14]; Prev* represents the latest fast method with parameters selected for adapting BDPSNR of approximately - 0.01dB. The changed parameters used in the test sequences are all

the same. HDA6 represents the HDA method when THD is 6. Similarly, Prop6 means the proposed method with the HDA6 method. LDPC TS and WZ TS indicate time savings respectively in LDPC decoder and total WZ decoder. *Timesaving(TS)* is calculated as,

$$Time\,Saving\,(TS) = \frac{Time_{original} - Time_{fast}}{Time_{original}} \times 100 \qquad (11)$$

In eq.(11), $Time_{original}$ is the decoding time in a conventional (LDPC or WZ) decoder and $Time_{fast}$ is the decoding time when fast LDPC decoding method is used.

Comparison of Prev and Prop6 in Table I shows little difference in time savings, but quite a large difference in BDPSNR. In comparing the results of Prev* and Prop6, Prop6 has a larger average time saving－the difference is about 12.04% for the LDPC decoding and 6.8% for the WZ decoding. Prop6 shows better R-D performance (BDPSNR) compared with Prev* as well. In Prop6, the best result is obtained from the Hall monitor sequence, which shows a time saving of 76.95% in LDPC decoding － it improves LDPC TS of 13.64% compared to Prev*. However, the complexity ratio the LDPC decoder accounts for in the WZ decoder is the smallest among the four test sequences; thus, it attains the smallest WZ time savings (30.15%) and difference of WZ time saving (5.82% compared with Prev*). The Stefan sequence shows the opposite results－the smallest difference of LDPC time saving (11.2% compared with Prev*) but the largest difference of WZ time saving (7.76% compared with Prev*), because the complexity ratio the LDPC decoder occupies in the WZ decoder is the largest among the four test sequences.

V. Conclusion

This paper proposed a fast LDPC decoding method with a negligible degradation of R-D performance. Compared with previous works, the proposed method shows much smaller loss in R-D performance but meaningful time savings in both the LDPC and WZ decoder (12.04% and 6.8%, respectively). The proposed method requires only one user-specified parameter (THD of HDA) so it sees less influence from parameters compared with previous works.

Even though we applied the proposed fast LDPC decoding method to WZ decoding, the complexity ratio of the LDPC decoder in the WZ decoder is still high. Therefore, a more accurate rate estimation method, which does not degrade R-D performance much, and an additional fast decoding algorithm, like the HDA method, is desired. These would further reduce the WZ decoder complexity while resisting the influence of sequence characteristics.

REFERENCES

[1] D. Slepian and J.K. Wolf, “Noiseless Coding of Correlated Information Source,” IEEE Transactions on Information Theory, vol. IT-19, no. 4, pp. 471-480. July 1973.

[2] A. Wyner and J. Ziv, “The Rate-distortion Function for Source Coding with Side Information at the Decoder,” IEEE Trans. on Information Theory, vol. 22, no. 1, pp. 1-10, July 1976.

[3] A. Aaron, R. Zhang, and B. Girod, “Wyner-Ziv Coding of Motion Video,” Asilomar Conference on Signals, Systems and Computers, pp. 240-244, Pacific Grove, CA, USA, November 2002.

[4] C. Berrou and A. Glavieux “Near Shannon limit error correcting coding and decoding: Turbo-codes,” Proc. of IEEE International Conference on Communications, 1993, vol.2, pp.1064-1070, Geneva, Switzerland, May 1993.

[5] D. J. C. Mackay, “Good error-correcting codes based on very sparse matrices,” IEEE Trans. on Info. Theory, vol. 45, no. 2, pp. 399-431, March 1999.

[6] C. Brites and F. Pereira, “Correlation noise modeling for efficient pixel and transform domain Wyner-Ziv video coding,” IEEE Trans. Circuits and Systems for Video Tech., Vol. 18, no. 9, pp. 1177-1190, September 2008.

[7] L. Lu, D. He, and A. Jagmohan, “Side information generation for distributed video coding,” Proc. of IEEE ICIP, vol. 2, pp. 13-16, San Antonio, USA, September 2007.
[8] B. Ko, H.J. Shim, and B. Jeon, “Wyner-Ziv Video Coding with Side Matching for Improved Side Information,” Proc. of PSIVT, LNCS, vol. 4872, pp. 816-825, December 2007.
[9] C. Park, H.J. Shim, and B. Jeon, “Multiple noise model in distributed video coding”, Proc. of IWAIT, pp. 154-157, Kuala Lumpur, Malaysia, January 2010.
[10] D. Varodayan, A. Aaron, and B. Girod, “Rate-adaptive codes for distributed source coding,” EURASIP Signal Processing Journal, Special Section on Distributed Source Coding, Vol. 86, no. 11, pp. 3123-3130, November 2006.
[11] F. Kienle and N. Wehn, “Low complexity stopping criteria for LDPC code decoders,” Proc. of IEEE Vehicular Technology Conference, vol. 1, pp. 606-609, Stockholm, Sweden, June 2005.
[12] C. Brites, and F. Pereira, “Encoder rate control for transform domain Wyner-Ziv video coding,” Proc. of the IEEE ICIP, vol. 2, pp. 5-8, San Antonio, TX, USA, September 2007.
[13] J.D. Areia, J. Ascenso, C. Brites, F. Pereira, “Low complexity hybrid rate control for lower complexity Wyner-Ziv video decoding,” Proc. of EUSIPCO, Lausanne, Switzerland, August 2008.
[14] J. Ascenso, F. Pereira, “Complexity efficient mechanism for LDPC based Wyner-Ziv video decoding,” Proc. of The fifth International Mobile Multimedia Communications Conference, London, United Kingdom, September 2009.
[15] R.Y. Shao, S. Lin, and M.P.C. Fossorier, “Two simple stopping criteria for turbo decoding,” IEEE Trans. Comm., vol. 47, no. 8, pp. 1117-1120, August 1999.
[16] C. Brites, J. Ascenso, and F. Pereira, “Improving transform domain Wyner-Ziv coding performance,” Proc. of IEEE ICASSP, pp. 525-528, Toulouse, France, May 2006.
[17] J. Ascenso and F. Pereira, “Design and performance of a novel low-density parity-check code for distributed video coding,” Proc. of IEEE ICIP, pp. 1116-1119, San Diego, USA, October 2008.
[18] D. Kubasov, K. Lajnef, and C. Guillemot, “A hybrid encoder/decoder rate control for a Wyner-Ziv video codec with a feedback channel,” Proc. of IEEE MMSP, pp.816-825, Crete, Greece, October 2007.
[19] A. D. Liveris, Z. Xiong, and C.N. Georghiades, “Compression of binary sources with side information at the decoder using LDPC codes,” IEEE Communication Letters. vol. 6, no. 10, pp. 440-442, October 2002.
[20] R. Westerlaken, S. Borchert, R.K. Gunnewiek, I. Lagendijk, “Analyzing symbol and bit plane-based LDPC in Distributed Video Coding,” Proc. of IEEE ICIP, vol. 2, pp. 17-20, San Antonio, USA, September 2007.
[21] X. Artigas, J. Ascenso, M. Dalai, S. Klomp, D. Kubasov, and M. Ouaret, “The DISCOVER codec: architecture, techniques and evaluation,” PCS, Lisbon, Portugal, November 2007.
[22] H.J. Shim, and B. Jeon, “Distributed Video Coding,” The Magazine of the IEIE, vol.36, no.4, pp. 91-105, April 2009.
[23] B. Ko, H.J. Shim, and B. Jeon, “Low Complexity Video Encoding Using Turbo Decoding Error Concealments for Sensor Network Application,” Journal of the Institute of Electronics Engineers of Korea, vol.45, no.1, pp. 11-21, January 2008.
[24] R. Oh, H.J. Shim, and B. Jeon, “Adaptive Hard Decision Aided Fast Decoding Method in Distributed Video Coding,” Journal of the Institute of Electronics Engineers of Korea, vol.47, no.6, pp. 66-74, November 2010.

저 자 소 개

오 양 근(학생회원)

2009년 성균관대학교 정보통신 공학부 졸업 (학사).
2011년 성균관대학교 정보 통신공학부 졸업 (석사)
2011년~현재 삼성전자 연구원
<주관심분야 : 멀티미디어 영상압축, 신호처리>

심 혁 재(학생회원)

2000년 성균관대학교 전자공학과 졸업 (학사).
2002년 성균관대학교 정보통신 공학부 졸업 (석사).
2013년 성균관대학교 정보통신 공학부 졸업 (공학박사)
2013년~현재 성균관대학교 정보통신공학부 박사 후연구원
<주관심분야 : 멀티미디어 영상압축, 신호처리>

전 병 우(정회원)

1985년 서울대학교 전자공학과 졸업 (학사).
1987년 서울대학교 전자공학과 졸업 (석사).
1992년 Purdue Univ, School of Elec. 졸업 (공학박사)
1993년~1997년 삼성전자 신호처리연구소 선임/수석연구원
1997년~현재 성균관대학교 정보통신공학부 교수
<주관심분야 : 멀티미디어 영상압축, 영상인식, 신호처리>