LL-FMC: Low-latency frame memory compression scheme with high reconstructed quality

Jin Shin, Jong Ho Lee, and Hyun Kim ✉

Department of Electrical and Information Engineering, Research Center for Electrical and Information Technology, Seoul National University of Science and Technology, Seoul, South Korea

✉ E-mail: hyunkim@seoultech.ac.kr

Frame memory compression (FMC) saves power by reducing the data bandwidth for accessing reference frames of video coding standards in the external memory. However, since the continuously accumulated data loss degrades the performance of video coding standards, the quality of the reconstructed reference frame must be improved. Previous lossy frame memory compression methods focused on processing speed or simplicity, but their performance remains limited in terms of the quality of the reconstructed frame. In this study, a Huffman coding with a fixed codebook designed to considerably reduce the latency caused by arithmetic coding in lossless compression schemes is proposed. A novel lossy frame memory compression technique based on the Hadamard transform that does not require additional memory access using a lookup table is also proposed. The result of experiments on several reconstructed videos preprocessed with HEVC show that the proposed method improves BDPSNR by 10.7 dB and BDBR by 40.4% compared with the DWT+SPIHT-based compression technique which is widely used for lossy frame memory compression.

Introduction: With the growth of the media platform market such as over-the-top media services, the importance of video coding standards (e.g. H.264 [1], HEVC [2], and VVC [3]) has been increasingly emphasized. To achieve a high compression ratio, these video codecs perform complex computations and continuously access a reference frame located in external memory for motion compensation in block units, which is the primary contributor to the power consumption of such methods. To solve this problem, numerous studies have considered frame memory compression (FMC) [4–6] methods designed to reduce the bandwidth between the codec and external memory. These techniques can be classified as lossy and lossless FMC.

Lossy FMC methods are being studied continuously in the direction of increasing peak signal-to-noise ratio (PSNR) due to the advantage of being able to significantly reduce memory bandwidth despite the issue that the reconstructed video quality is continuously degraded due to the accumulated error of the reference data. A trade-off between compression ratio and video quality must be carefully considered in the design of lossy FMC methods, and they commonly implement a combination of transform-based and quantization algorithms. In particular, the discrete wavelet transform (DWT) [7] can perform multi-level decomposition and is used together with set partitioning in hierarchical tree algorithm (SPIHT) [8] to accurately control the target bit length for each block (i.e. progressive coding) and to retain the quality of the reconstructed frame. In addition, the combination of DWT and SPIHT significantly improved PSNR with minimal computational resources through dynamic allocation [5, 9]. However, a minimum of three levels of decomposition are required to take full advantage of SPIHT, and its compression performance is relatively poor if this decomposition level is not met.

In contrast, lossless FMC methods [6] can achieve high quality but require relatively high memory bandwidth and exhibit high computational complexity. As an example of conventional transform-based lossless compression schemes, the Hadamard transform (HT) [10] and entropy coding (i.e., Huffman coding [11]) can be used for FMC, but they occur a delay due to the preprocessing for the codebook, and consequently, efficient FMC design supporting real-time operation is impossible with these methods. To compensate for this limitation, Yng et al. [4] maintained lower complexity by using a modified Hadamard transform and this method does not require a pixel rearrangement process, thereby reducing temporary memory and running time compared to 2D-DWT [7]. However, this method has the limitation of compressing the AC coefficient by applying the relatively complex adaptive Golomb–

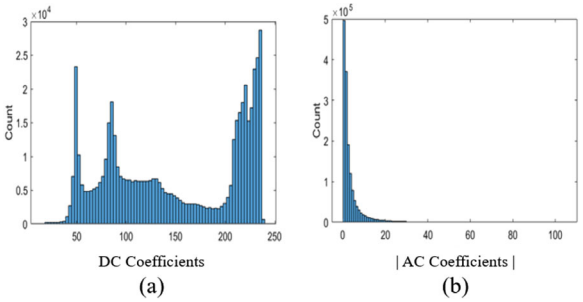

Fig. 1 *Histogram of coefficients measured on a single frame of BQ_Terrace_8bits. (a) DC coefficients; (b) AC coefficients.*

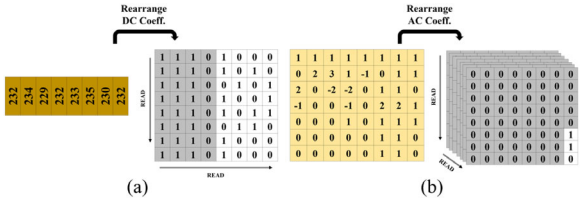

Fig. 2 *Rearrangement technique for fixed codebook Huffman. (a,b) The bit-plane of the DC and AC coefficients, respectively. The rearranged streams are read from top to bottom from left to right and appended to the bitstream.*

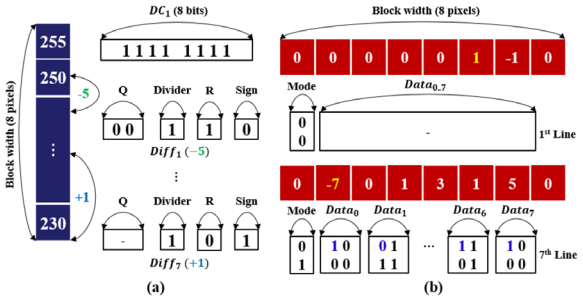

Fig. 3 *Overview of the proposed entropy coding. (a,b) The bitstreams of the DC and AC coefficients, respectively.*

Rice (GR) algorithm. In addition, a mixed lossy and lossless algorithm was designed to improve compression performance and reduce memory bandwidth by considering the existing fast algorithms. However, this approach cannot reliably achieve high throughput because frame recompression access requires sequential differential pulse code modulation (DPCM) processing for all pixels.

In this study, we focus on reducing computational costs and maximizing compression performance for efficient block-wise compression. We propose compression techniques to comprehensively solve the problems described above, and the contributions of this study are summarized as follows:

- To maximize the benefits of HT-based compression, we separate AC coefficients into bit-plane units and leverage the very high probability that bits located close to the most significant bit (MSB) have a value of zero to increase compression efficiency.
- To solve the adaptive codebook problem which has the greatest impact on the complexity of Huffman coding, we design a fixed codebook-based Huffman coding using a new bit-plane rearrangement technique (RT) for AC coefficients.
- Finally, we present a method to completely eliminate temporary memory, bit depth, and many look-up tables to store values in bit-plane units in a fixed codebook-based Huffman coding.

Our proposed method allows parallel processing of AC coefficients without additional computation except checking MSB bit positions by the block line. Consequently, the proposed compression technique can be considered as a practical approach because it can achieve both lower computational complexity and superior performance compared to con-

Table 1. Comparison of BDPSNR and BDBR with the proposed entropy coding.

	BDPSNR (db)			BDBR (%)		
Sequences	DWTSPIHT [12]	DBA [5]	Fixed code-book Huff.	DWTSPIHT [12]	DBA [5]	Fixed code-book Huff.
4K HDR	4.09	3.55	0.95	-13.28	-8.77	-7.06
Eclipse	1.35	0.39	0.15	-3.06	-0.29	-1.47
BQ Terrace-10	6.8	3.97	0.76	-19.18	-13.14	-5.46
BQ Terrace-8	18.05	13.85	4.76	-39.22	-36.22	-28.48
Bosphorus	22.58	22.58	22.58	-79.55	-31.34	-33.09
ReadySteadyGo	11.47	7.76	3.08	-87.99	-21.66	-28.92
Average	10.72	6.65	2.1	-40.38	-18.57	-17.41

Table 2. Comparison of running time (ms) in BQ terrace-8 bits.

DWT	HT	SPIHT	Fixed codebook Huff.	Proposed
33.2 ms	6.68 ms	27.88 ms	4.98 ms	2.83 ms

ventional DWT and discrete cosine transform (DCT)-based lossy compression methods.

Prior works:

Lossy FMC schemes with SPIHT: The SPIHT algorithm has been widely used in transform-based FMC schemes where efficient lossy coding is essential because it does not require arithmetic operation and can achieve a hardware-friendly design that can be easily parallelized [12]. In general, DWT decomposes an image into a low-frequency bandwidth (i.e. LL) and three high-frequency bandwidths (i.e. LH, HL, and HH), which have a hierarchical tree structure called spatial orientation tree (SOT). SPIHT compresses frames in bit-plane units by taking advantage of this characteristic of DWT. DWT is most effective when decomposed into at least three levels. However, if either the width or height of the macroblock is less than 8 in an asymmetric form (e.g. 16×4, 32×4, 64×4), three-level decomposition is not possible. In contrast, because the DCT does not fundamentally have an SOT structure, it is decomposed into a DCT in units of 4×4 blocks, and each coefficient is grouped into four bandwidths according to its location for compensating its structural limitations. This strategy can transform asymmetric blocks that interfere with 3-level decomposition into symmetrical forms to improve the quality of the reconstructed images. Meanwhile, Kim et al. [5] presented a solution to address the potential limitations in which all blocks are allocated at the same compression ratio, even though SPIHT is able to generate a bitstream exactly at the target bit length. In detail, they used the relative cost obtained by counting the high-frequency coefficients decomposed by DWT per block, and complex blocks with high cost are encoded at aggressive compression ratios, whereas simple blocks with low cost are encoded at conservative compression ratios.

Hadamard transform-based lossless FMC schemes: In general, the Hadamard transform (HT) is used to design lossless or near-lossless FMC with entropy coding (e.g. variable length coding and arithmetic coding) owing to its less DC coefficient, relatively small variance for AC coefficients, and low computational complexity compared to DWT [12]. Nevertheless, the delayed nature of entropy coding is a drawback when implementing hardware design. As an example, in contrast to progressive coding such as SPIHT, Huffman coding [11] requires continuous updating of a lookup table codebook to achieve optimal compression performance. Hence, applying these methods to real-time applications is difficult owing to their variable latency. The GR [13] coding algorithm is widely used with DPCM as a representative lossless compression method, and encoded coefficients can be expressed as follows:

$$Enc_n = (c_n - c_{n+1})/2^k + R \quad (1)$$

where C_n denotes the n-th coefficient of the image block decomposed by HT, and the difference with the next coefficient C_{n+1} is divided by 2^k. Subsequently, the quotient is encoded by allocating bits by size, and the remaining R is encoded using a GR code consisting of a prefix code. This process is only performed for the AC coefficient. GR has relatively low computation complexity and does not require an update of the codebook, but the latency increases proportionally with the block size because the entire AC coefficients must be performed sequentially for the DPCM. In addition, the quality can be improved by adaptively selecting k, but the performance is limited in terms of retaining the quality of the original content because the block size is not large for the purpose of FMC.

Proposed methods: In this section, we propose a novel FMC technique that can perform lossless compression in a manner similar to that of lossy compression, using only the advantages of each of the above-mentioned coding methods including the HT. It should be noted that the HT is basically lossless, like DWT; however, in the proposed approach, mantissa bits are removed and eventually compressed with some loss.

Rearrangement technique for maximizing the efficiency of entropy coding: HT-based Huffman coding generates an adaptive code tree based on the frequency of each coefficient, which introduces a delay in execution time. In this subsection, we introduce a fixed codebook with the RT to circumvent this issue. The raw data (8×8) from the HT operation can be divided into low-frequency DC (1×8) and high-frequency AC (7×8) subbands. Figures 1a and 1b, respectively, represent the histograms of the DC and AC coefficients in a single frame of an 8-bit depth BQ-Terrace image [14]. In Figure 1b, AC coefficients are fundamentally different from DC coefficients, as they consist mainly of values close to "0," accounting for 31.94% of the frequency. To further accentuate this tendency, we introduce an RT that efficiently organizes the order of the bitstream before the prefix code encoding by dividing the coefficients into bit-depth units. The conventional approach appends to the bitstream in sequential order from the most significant bit (MSB) to the least significant bit (LSB) according to the coefficients. On the other hand, the RT method can increase the proportion of "0" by appending from the MSB in a bit-level order across all coefficients. Figure 2b illustrates the bit plane of the AC coefficients, where the last line carries two valid bits. Under the conventional approach, we could consider there being six "0"s and two "1"s, but with the proposed RT, we can achieve seven "0"s and one "1." By employing the RT, we have verified a 7.63% increase in the frequency of "0" in the BQ-Terrace image. Similarly, in Figure 2a, most bits near the MSB for DC coefficients are valid bits, and upon applying the RT, "1" has the second-highest frequency. This dramatic increase in the frequency of the "0" and "1" coefficients suggests the feasibility of using a fixed codebook rather than obtaining the codebook adaptively. We utilize the conventional Huffman code tree to generate unique prefix codes, further amplifying the compression efficiency by reducing the significantly prevalent "0" coefficients to a single bit. However, the constraint of expanding the tree in a one-way direction leads to a larger deviation in code length. Specifically, even "1," the second most frequent coefficient, is compressed to 5 bits (= 10110), and most coefficients are compressed to 9 bits on average. Nevertheless, the dominant frequency of "0" can compensate for this larger deviation in code length.

Implementation of the proposed entropy coding: The method introduced above does not require a codebook to be regenerated in certain periods, but still has the disadvantage of needing to be changed depending on the block size or bit depth. As a solution, we propose a method that enables parallel processing with no additional computational requirements other than searching for MSB bit positions by line. In this method, different compression strategies are adopted for the DC and AC coefficient. As shown in Figure 1a, the DC coefficients may be assigned different values for each block because the values are flexible within a single frame; however, they have similar values for adjacent pixels. These characteristics are highly suitable for adapting the DPCM-GR algorithm [4] to DC coefficients. Figure 3a illustrates a bitstream encoding DC coefficients with DPCM-GR, where k is fixed to increase the execution speed considering that the difference between the DC coefficients is sufficiently small. k is a parameter used as an exponent of 2 in (1) and adaptively determines the length of Q, a component of Enc_n. The first DC coefficient is preserved in the bitstream without considering the sign bit, where 1, R, and S denote the delimiter, remainder, and sign bit, respectively. The AC coefficients also exhibit high similarities between adjacent coefficients in the horizontal direction, and many values are either zero or close to zero, as mentioned above. Therefore, we only calculate the position of the MSB by line and support four modes (i.e., 00,01,10, and 11) using only two bits, as shown in Figure 3b. Although 14 ($=2 \times 7$) bits are inevitably allocated only to the AC coefficient when the block size is 8×8, if the mode is "00," the magnitude bits do not need to be included in the bitstream; thus, a significant compression effect can be achieved with this mode. The mode is determined by the position of the most significant valid bit in the largest coefficients among the seven coefficients included in the line (i.e., [0,1] → "00," [2,3] → "01," [4,5] → "10," and [6,7] → "11"), and all the coefficients on the line are encoded in the same size according to this mode value. Thus, parallel processing can be performed because each line of AC coefficients uses the same instruction. The second proposed method can improve compression performance by considering compression ratio and reconstruction quality while eliminating processes that can delay processing speed as much as possible.

Experimental results: In this section, we present the results of comparative experiments between our proposed lossy FMC technique and existing lossy/lossless FMC techniques. All experiments were designed as FMC environments in which reconstructed images in HEVC were recompressed in blocks with a size of 16×4. These sequences [14, 15] were reconstructed using the common test conditions of HEVC (i.e. Profile: Main profile, QP: 22. Entropy coding: CABAC, YUV420 format).

Table 1 shows the results of a fair comparison of the compression performance of the Y components of each compression method using BDPSNR and BDBR [16]. It should be noted that BDPSNR and BDBR represent the performance gaps between the two methods, and we present the BDBR and BDPSNR of the remaining three methods using the proposed entropy coding as an anchor. The experimental results show the proposed entropy coding improved BDPSNR by 10.72, 6.65, and 2.1 dB, and reduced BDBR by 40.38%, 18.57%, and 17.41%, respectively, compared to DWT-SPIHT [12], DBA [5], and fixed-codebook Huffman.

In Table 2, the running times of the transform and compression techniques applied in the experiment are presented, which are related to computational complexity. We present the average value in milliseconds after measuring five times per method using only a single CPU core for highly reliable comparisons. The main reason that the running time of DWT was longer than that of HT is that the coefficients must be rearranged for the SOT. In addition, the two proposed methods require significantly fewer computations than SPIHT because they involve a degree of complexity in referring to the codebook or extracting the position of the MSB. The method with the fixed codebook Huffman is slower than the proposed approach because it constantly allocates memory from the rearrangement process to sequentially aggregate the decomposed coefficients of the HT from the MSB. Consequently, the proposed method ($=6.68+2.83$) achieved a running time 6.42 times faster than the conventional DWT-SPIHT compression method ($=33.2+27.88$) [12].

Conclusion: In this study, we have introduced a method to dramatically reduce latency by fixing the codebook required for Hadamard transform-based Huffman coding and novel lossy FMC techniques that do not require a codebook. The computational complexity of the proposed methods is lower than that of DWT-SPIHT, which is frequently used for lossy FMC. This advantage enables parallel processing with a short delay time and improves the quality of reconstructed frames.

Acknowledgments: This work was partly supported by the National Research Foundation of Korea (NRF) grant funded by the Korea government (MSIT) (No. 2022R1F1A1062786) and the Basic Science Research Program through the National Research Foundation of Korea (NRF) funded by the Ministry of Education under Grant NRF-2019R1A6A1A03032119.

Data availability statement: Data are derived from public domain resources.

Received: *29 March 2023* Accepted: *12 July 2023*
doi: 10.1049/ell2.12893

References

1 Wiegand, T., et al.: Overview of the H. 264/AVC video coding standard. *IEEE Trans. Circuits Syst. Video Technol*. **13**(7), 560–576 (2003)
2 Sullivan, G.J., et al.: Overview of the high efficiency video coding (HEVC) standard. *IEEE Trans. Circuits Syst. Video Technol*. **22**(12), 1649–1668 (2012)
3 Fu, T., et al.: Fast CU partitioning algorithm for H. 266/VVC intra-frame coding. In: *2019 IEEE International Conference on MMultimedia and Expo (ICME)*, pp. 55–60. IEEE, Piscataway, NJ (2019)
4 Yng, T.L.B., Lee, B.G., Yoo, H.: A low complexity and lossless frame memory compression for display devices. *IEEE Trans. Consum. Electron.* **54**(3), 1453–1458 (2008)
5 Kim, H., No, A., Lee, H.J.: SPIHT algorithm with adaptive selection of compression ratio depending on DWT coefficients. *IEEE Trans. Multimedia* **20**(12), 3200–3211 (2018)
6 Lian, X., et al.: Lossless frame memory compression using pixel-grain prediction and dynamic order entropy coding. *IEEE Trans. Circuits Syst. Video Technol*. **26**(1), 223–235 (2015)
7 Grangetto, M., et al.: Optimization and implementation of the integer wavelet transform for image coding. *IEEE Trans. Image Process*. **11**(6), 596–604 (2002)
8 Dragotti, P.L., Poggi, G., Ragozini, A.R.: Compression of multispectral images by three-dimensional SPIHT algorithm. *IEEE Trans. Geosci. Remote Sens.* **38**(1), 416–428 (2000)
9 Shin, J., Kim, H.: RL-SPIHT: reinforcement learning-based adaptive selection of compression ratios for 1-D SPIHT algorithm. *IEEE Access* **9**, 82485–82496 (2021)
10 Pratt, W.K., Kane, J., Andrews, H.C.: Hadamard transform image coding. *Proc. IEEE* **57**(1), 58–68 (1969)
11 Hashemian, R.: Memory efficient and high-speed search huffman coding. *IEEE Trans. Commun.* **43**(10), 2576–2581 (1995)
12 Jin, Y., Lee, H.J.: A block-based pass-parallel SPIHT algorithm. *IEEE Trans. Circuits Syst. Video Technol.* **22**(7), 1064–1075 (2012)
13 Kim, H.S., et al.: A lossless color image compression architecture using a parallel Golomb–Rice hardware CODEC. *IEEE Trans. Circuits Syst. Video Technol*. **21**(11), 1581–1587 (2011)
14 Nguyen, T., Marpe, D.: Future video coding technologies: a performance evaluation of av1, jem, vp9, and hm. In: *2018 picture coding symposium (PCS)*, pp. 31–35. IEEE, Piscataway, NJ, (2018)
15 Lemmetti, A., et al.: Kvazaar 2.0: fast and efficient open-source HEVC inter encoder. In: *Proceedings of the 11th ACM Multimedia Systems Conference*. ACM, New York (2020). https://doi.org/10.1145/3339825.3394927
16 Bjontegaard, G.: Calculation of average PSNR differences between RD-curves. ITU SG16 Doc. VCEG-M33 (2001)
17 Avcibas, I., et al.: A progressive lossless/near-lossless image compression algorithm. *IEEE Signal Process Lett*. **9**(10), 312–314 (2002)