Parallel Content-Aware Adaptive Quantization-Oriented Lossy Frame Memory Recompression for HEVC

Xiaocong Lian, *Student Member, IEEE*, Zhenyu Liu, *Member, IEEE*, Wei Zhou, *Member, IEEE*, and Zhemin Duan

***Abstract*— Since the development of ultrahigh-definition video, the huge bandwidth and power requirements of external memory have hindered the development of video encoder applications. Power constraints have become a particularly serious problem for portable video codec systems. With high-rate configurations [quantization parameter (QP) $\leq$ 22 in HEVC test model (HM) reference software], the compression performance of the existing lossless compression algorithms noticeably degrades, because the reference frames are becoming rich of textures. On the other hand, the mathematical analysis of this paper revealed that more quantization noises can be endured by the texture-rich area. Therefore, we develop an adaptive quantization-oriented parallel lossy frame memory recompression algorithm. The contributions of this paper include the following. First, a content-aware adaptive quantization method is devised to achieve a stable high compression ratio that does not deteriorate for highly quality texture-rich pictures. When QP$\in$ [12, 22], a data reduction ratio improvement of up to 14% is obtained compared with the best lossless algorithm. Furthermore, it can reduce the quality loss by 0.49–3.36 dB in terms of Bjøntegaard delta peak signal-to-noise rate (BD-PSNR) compared with the fixed length quantization method. Second, to solve the low throughput problem caused by the pixel-grain prediction method, a parallel directional prediction scheme is developed. It can double or quadruple the throughput with a prediction accuracy loss of only 1.7% or 3.3%, respectively. Using the above-mentioned methods, bandwidth and memory requirements are reduced up to 70.6% and 41.0%, respectively, with a corresponding savings of 59.3% in the dynamic power consumption of the off-chip dynamic random access memory, while the BD-PSNR is −0.04 dB, or, equivalently, Bjøntegaard delta bit rate (BD-BR) is 1.27%. Using TSMC 65-nm CMOS technology, the proposed frame memory compressor and decompressor can achieve the throughputs of up to 2.89 and 2.26 Gpixels/s, respectively. It is applicable to a Super Hi-Vision(8K)@68-frames/s real-time encoding with a Level D reference data reuse scheme.**

***Index Terms*— Frame memory recompression (FMR), High Efficiency Video Coding (HEVC), lossy compression, low power.**

I. Introduction

SINCE the beginning of the 21st century, video related fields have been developing rapidly. Video definition has dramatically increased, and new Quad Full High Definition (4K) and Super Hi-Vision (SHV, 8K) applications are appearing all the time. The developments in the video field require a new video coding standard with higher coding efficiency to support the development of the higher definition video. Therefore, High Efficiency Video Coding (HEVC) [1] was developed by the Joint Collaborative Team on Video Coding. The HEVC can save the bitrate by 40%-50% compared to H.264/AVC [2], especially for Ultra-High-Definition video.

Off-chip dynamic random access memory (DRAM) [3] is used as frame memory due to its high integration density. Bandwidth and power requirements are the two obstacles in reference frame storage. By applying the data reuse scheme, the bandwidth limitation can be ameliorated [4]. For example, by applying the Level D data reuse scheme [5], the current double-date-rate three (DDR3) technology meets the bandwidth requirements of an 8K@60-frames/s video encoding with a unique motion estimation reference frame. The power requirements of DRAM have become a serious bottleneck in encoder design, especially for portable video applications. The dynamic power used by the DRAM consists of three primary components, the internal read/write power, the IO terminal power, and the activate power, which occupy 40.1%, 21.7%, and 38.2% of the total dynamic power, respectively [6]. Power can be reduced in the first two components when the reference frame pixels are compressed. In contrast, activate power reduction depends on the higher utilization of a row buffer in the DRAM, which requires the reference frame compressor to provide a compression ratio of no less than 50%.

To be compatible with the very large scale integration (VLSI) design, the reference frame compression algorithm must possess the properties of low complexity and high throughput. Intensive computational complexity and, particularly, high processing latency hinder the application of traditional compression algorithms [7]. Extensive frame memory recompression (FMR) schemes can be classified in two

Manuscript received July 7, 2016; revised October 15, 2016 and November 13, 2016; accepted December 5, 2016. Date of publication December 13, 2016; date of current version April 4, 2018. This work was supported in part by the National Natural Science Foundation of China under Grant 61602383 and Grant 61772424, in part by the Fundamental Research Funds for the Central Universities under Grant 3102016ZY024, in part by the Fundamental Natural Science Research Funds of Shaanxi Province under Grant 2016JM6077, and in part by the Innovation Foundation for Doctor Dissertation of Northwestern Polytechnical University under Grant CX201616. This paper was recommended by Associate Editor G. Valenzise. *(Corresponding author: Wei Zhou.)*

X. Lian, W. Zhou, and Z. Duan are with the School of Electronics and Information, Northwestern Polytechnical University, Xi'an 710129, China (e-mail: lianxiaocong123@mail.nwpu.edu.cn; zhouwei@nwpu.edu.cn; zhemind@nwpu.edu.cn).

Z. Liu is with the Tsinghua National Laboratory for Information Science and Technology, Research Institute of Information Technology, Tsinghua University, Beijing 100084, China (e-mail: liuzhenyu73@mail.tsinghua.edu.cn).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TCSVT.2016.2638857

main categories: lossless and lossy FMR schemes [8]–[15]. A lossless image compression architecture is proposed in [10] that consists of differential–differential pulse coded modulation (DDPCM) and Golomb–Rice coding. It achieves a data reduction ratio (DRR) of 60.3%. A multimode DPCM and averaging prediction & semifixed length entropy coding (MDA & SFL) algorithm is proposed in [11], with an achieved DRR as high as 61.9%. On the other hand, lossy reference frame recompression algorithms capitalize on the insensitivity of human eyes to small quality loss to further improve the compression efficiency. Lee *et al.* [12] achieve a DRR of no less than 50% at the cost of a 1.03-dB image quality degradation. A mixed lossy & lossless (MLL) algorithm is proposed in [13]. The lossy data are used only for integer motion estimation (IME), while the original data will be reconstructed and used for fractional motion estimation (FME) and motion compensation (MC). The proposed MLL process of [13] results in a −0.01-dB quality change in PSNR. However, the truncated bits of the best match image block must be retrieved from the external DRAM during FME processing. This retrieval from the DRAM introduces long latency between IME and FME, i.e., degradation of the overall encoder throughput.

In our previous work [16], a lossless frame memory compression algorithm consisting of pixel-grain prediction and dynamic order unary/Exp-Golomb coding is proposed. However, the algorithm in [16] has the shortages of the unstable compression ratio and the low processing throughputs. In particular, under the conditions of complex texture or high-rate configurations, the amplitudes of the prediction residues in the reference frames increase significantly and this behavior will decrease the compression efficiency of our previous lossless algorithm. In addition, due to data dependence, the prediction speed of the previous compressor/decompressor is limited. The maximum throughput of the previous decompressor is merely 0.78 Gpixels/s.

Lossy compression algorithms apply a quantization mechanism to reduce the entropy of prediction residues. The quantization mechanisms in the existing lossy compression algorithms are divided primarily into two categories: a fixed length quantization mechanism and a target compression ratio-oriented quantization mechanism [8], [12]. The quantization on reference frames produces the additional noises, which always degrade the coding quality of HEVC encoder. The limitation of previous works stems from the indiscriminate quantization strategy, which ignores the impact of picture texture complexity on the coding quality loss of the HEVC encoder. In fact, our mathematical analysis in Section II-A will reveal that highly complex textures can tolerate more noises than the homogeneous counterparts.

In this paper, we explore an image-context-oriented dynamic quantization scheme for the lossy FMR to further improve the reference frame compression efficiency, on the premise that the overall encoding quality is maintained. In comparison to our previous study [16], the contributions of this paper include the following. We first design a content-aware adaptive quantization method to further reduce the entropy of prediction residues. This method can choose an appropriate reference

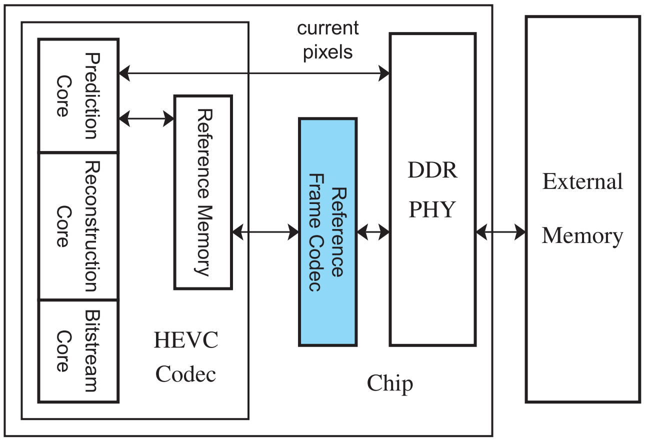


Fig. 1. High-level block diagram for the proposed compression system.

quantization length (RQL) for an image block based on its original entropy quantity, and thus reduce the relative coding quality loss of our lossy reference frame compression. A stable high compression ratio and memory reduction ratio that are not deteriorated by the highly quality texture-rich pictures can be achieved. Second, we develop a parallel directional prediction scheme to overcome the low throughput hindrance of our original pixel-grain directional prediction method. Third, in the VLSI design, we develop the lookup table circuits to implement the logarithm-based adaptive quantization algorithm, which consequently reduces hardware cost by 45.3 k-gate and improves the maximum frequency by 425 MHz.

The rest of this paper is organized as follows. The proposed lossy FMR algorithm is described in Section II. The VLSI implementation of the compression and decompression modules is explained in Section III. Section IV presents the experimental results. Finally, the conclusions are given in Section V.

II. Lossy Frame Memory Recompression

The reference pixels are used in the IME, FME, and MC modules of the HEVC encoder. Reading the reference frame has large bandwidth and power requirements, especially for a multiple-reference-frame scheme. The proposed FMR scheme aims to efficiently reduce the external memory bandwidth and power requirements. When encoding low-resolution video, small processing units are helpful in allowing for flexibility in the choice of search window. Therefore, most existing FMR algorithms adopt an 8 × 8 or 16 × 16 processing unit. For the current DDR3, the IO width is 64 b, and the burst length (BL) is 4 or 8. As the original 8 × 8 8-b pixel block is 512 b, only a single burst is needed to transmit the uncompressed data when the BL is 8. Therefore, the performance of an algorithm using an 8 × 8 processing unit cannot be effectively used to reduce the IO bandwidth. Based on the above-mentioned analysis, the basic luma partition size is defined as 16 × 16, and the corresponding chroma partition as 8 × 8 with a 4:2:0 sampling format.

The high-level block diagram of our compression system is shown in Fig. 1. The proposed reference frame codec is responsible for the compression and the decompression of reference frame image blocks. Section II-A describes the proposed content-aware adaptive quantization mechanism. The parallel directional prediction scheme is explained in Section II-B. Section II-C illustrates the dynamic kth-order unary/Exp-Golomb coding method. Finally, the partition group (PG) table-based compression storage scheme is provided in Section II-D.

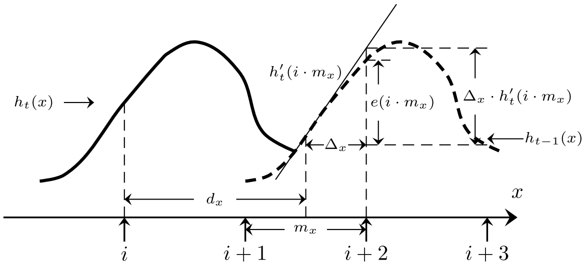


Fig. 2. Analysis of prediction residues based on edge intensity in the 1D domain (i, $i+1$, $i+2$, and $i+3$ are the camera sensors).

A. Content-Aware Adaptive Quantization Mechanism

In this section, we describe the content-aware quantization algorithm. We first investigate the effect of quantization noise stemming from lossy reference frame compression on the HEVC coding quality. Next, we provide the dynamic RQL scheme for the given coding quality constraints.

Let $r(u, v)$ represent the discrete 2D DCT transform coefficients of the prediction residues in the HEVC coding, and $r(u, v)$ is identified as a memoryless signal. The discrete cosine transform (DCT) coefficients are generally modeled as Laplacian distribution [17]. The literature [18] provided the approximated rate–distortion (R-D) function of Laplacian source with respect to the mean squared error criterion at the low distortion. However, the universal R-D analytical expression of Laplacian source with respect to the mean squared error criterion is not obtained. On the other hand, the Gaussian distribution is a proper approximation to the Laglacian one, and furthermore, the explicit R-D function of Gaussian source with respect to the mean squared error criterion has been derived in [19]. In consequence, many studies [20], [21] applied the Gaussian distributed DCT coefficients model to do the quantitative analysis.

Based on the discrete stationary Gaussian source R-D function of R-D theory [19], the distortion D and the corresponding rate R of each transform coefficient with respect to the mean squared error criterion can be expressed as

$$\begin{cases} D = \min(\sigma, \mathscr{S}_{\rm rr}(u, v)) \\ R = \max\left(0, \dfrac{1}{2}\log_2 \dfrac{\mathscr{S}_{\rm rr}(u, v)}{\sigma}\right) \end{cases} \tag{1}$$

where $\mathscr{S}_{\rm rr}(u, v)$ is the power spectral density of $r(u, v)$. σ is the quantization noise, which has the formulation $\sigma = Q^2/12$ (Q is the quantization step in the HEVC encoder).

In our lossy reference frame recompression algorithm, the RQL is l ($l \in \{0, 1, 2, 3\}$). That is, the lowest l bits of the prediction residues (ϵ), which are generated by the parallel directional prediction (to be described in Section II-B), are discarded. In consequence, additional noise will be introduced to the transformed prediction residues. Let $\Delta_{\rm rr}$ denote the power spectral density of the noise from quantization in lossy reference frame compression, and the corresponding distortion and rate are formulated as

$$\begin{cases} \widetilde{D} = \min(\sigma, \mathscr{S}_{\rm rr}(u, v) + \Delta_{\rm rr}) \\ \widetilde{R} = \max\left(0, \dfrac{1}{2}\log_2 \dfrac{\mathscr{S}_{\rm rr}(u, v) + \Delta_{\rm rr}}{\sigma}\right). \end{cases} \tag{2}$$

There are two cases for the effect of $\Delta_{\rm rr}$ [22]. First, if $\mathscr{S}_{\rm rr}(u, v) + \Delta_{\rm rr} < \sigma$, $\Delta_{\rm rr}$ will not have any adverse effect on the rate cost. Second, if $\mathscr{S}_{\rm rr}(u, v) + \Delta_{\rm rr} \geq \sigma$ and $\mathscr{S}_{\rm rr}(u, v) \gg \Delta_{\rm rr}$, with Taylor series, the difference in rate cost ($d\widetilde{R}$) can be described as $d\widetilde{R} = \dfrac{\Delta_{\rm rr}}{\mathscr{S}_{\rm rr}(u, v)}$, where $\Delta_{\rm rr} = 2^{2l}/\alpha$ (α is a constant value). If $d\widetilde{R}$ is assumed to be equal to βR, based on the above-mentioned equations, we can deduce the value of l ($0 \leq l \leq 3$) as

$$\begin{cases} 0 & \mathscr{S}_{\rm rr}(u, v) < \sigma \\ \dfrac{1}{2}\log_2\left(\gamma\, \mathscr{S}_{\rm rr}(u, v) \log_2 \dfrac{\mathscr{S}_{\rm rr}(u, v)}{\sigma}\right) & \mathscr{S}_{\rm rr}(u, v) \geq \sigma \end{cases} \tag{3}$$

where $\gamma = \alpha \cdot \beta/2$. For one 16×16 partition, we get one value of l. l is obtained in the encoding procedure and stored as auxiliary information for decoding. It can be observed that l increases with the prediction residue power $\mathscr{S}_{\rm rr}(u, v)$. Therefore, the goal of this paper is to determine l from the estimation of $\mathscr{S}_{\rm rr}(u, v)$. γ is defined based on experimentation, which will be described in Section IV. The value of $\mathscr{S}_{\rm rr}(u, v)$ depends on the future motion estimation using the current block as the reference. That is, we cannot derive the precise value of $\mathscr{S}_{\rm rr}(u, v)$ when compressing the current block. However, we can estimate the amplitude of $\mathscr{S}_{\rm rr}(u, v)$ from the textures and motions of the current block. Based on Parseval's theorem, we get

$$\sum_{u=0}^{15}\sum_{v=0}^{15} \mathscr{S}_{\rm rr}(u, v) = \sum_{i=0}^{15}\sum_{j=0}^{15} e^2(i, j) \tag{4}$$

where $e(i, j)$ is the prediction residue. We analyze the impact of edge intensity on the prediction residues in the 1D domain, as shown in Fig. 2. $h_t(x)$ and $h_{t-1}(x)$ denote the signals at time instances t and $t-1$, respectively, and the motion distance is d_x. The spatial-continuous signals are sampled by the camera sensors and the sampling interval is denoted as m_x. As seen in Fig. 2, the prediction residue $e(i \cdot m_x)$ of pixel i can be approximated as

$$e(i \cdot m_x) \approx \Delta_x \cdot h'_t(i \cdot m_x) \tag{5}$$

where $h'_t(i \cdot m_x)$ is the edge gradient of $h_t(x)$ at the ith camera sensor and Δ_x is the displacement estimation error

$$\Delta_x = d_x - \text{round}(d_x/m_x) \cdot m_x. \tag{6}$$

Therefore, when blocks have more complex textures and motions, the value of $\mathscr{S}_{\rm rr}(u, v)$ is increased accordingly. Based on the above-mentioned analysis, we have divided the

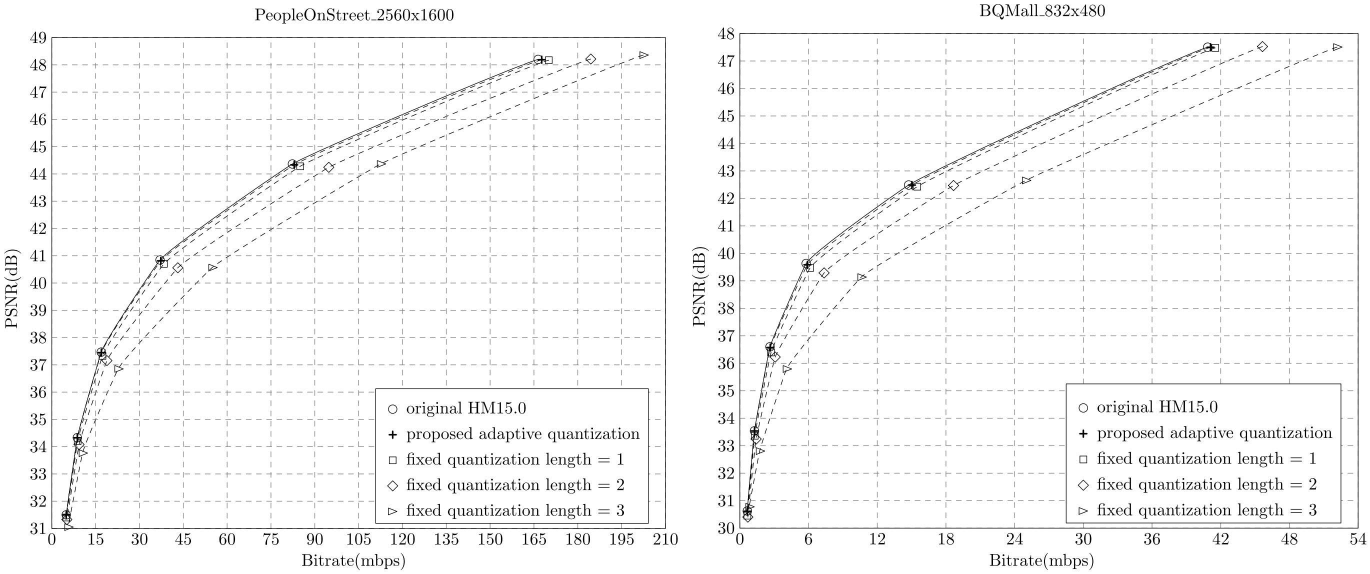


Fig. 3. R-D curve comparisons for the proposed content-aware adaptive and fixed length quantization mechanisms (simulation conditions: QP = 12, 17, 22, 27, 32, 37).

16 × 16 block into 64 2 × 2 subblocks, of which the edge vectors are calculated as follows:

$$\begin{cases} dx_{i,j} = p_{2i,2j+1} + p_{2i+1,2j+1} - p_{2i,2j} - p_{2i+1,2j} \\ dy_{i,j} = p_{2i+1,2j} + p_{2i+1,2j+1} - p_{2i,2j} - p_{2i,2j+1} \end{cases} \tag{7}$$

where $p_{i,j}$ is the pixel value, and $dx_{i,j}$ and $dy_{i,j}$ represent the edge strength in the vertical and horizontal directions of a 2 × 2 block, respectively.

The motion vectors ($\overrightarrow{mv}_{i,j} = \{mvx_{i,j}, mvy_{i,j}\}$) represent the motions of every 2 × 2 block. Meanwhile, to strengthen the effect of the motion vector, we introduce two variables Θ_x and Θ_y, which are defined as follows:

$$\begin{cases} \Theta_x = (|mvx_{i,j}| > 8)?2:1 \\ \Theta_y = (|mvy_{i,j}| > 8)?2:1. \end{cases} \tag{8}$$

Therefore, the approximate value of $\mathscr{S}_{\text{rr}}(u, v)$, i.e., $\widetilde{\mathscr{S}}_{\text{rr}}(u, v)$, is expressed as

$$\widetilde{\mathscr{S}}_{\text{rr}}(u, v) = \frac{1}{64} \sum_{i=0}^{7} \sum_{j=0}^{7} \left(dx_{i,j}^2 \left[\frac{\operatorname{mod}(mvx_{i,j}, 4)}{4} \right]^2 \Theta_x + dy_{i,j}^2 \left[\frac{\operatorname{mod}(mvy_{i,j}, 4)}{4} \right]^2 \Theta_y \right). \tag{9}$$

From (3) and (9), we can calculate l as

$$l = \begin{cases} 0 & \widetilde{\mathscr{S}}_{\text{rr}}(u, v) < \sigma \\ \dfrac{1}{2} \log_2 \left(\gamma \widetilde{\mathscr{S}}_{\text{rr}}(u, v) \log_2 \dfrac{\widetilde{\mathscr{S}}_{\text{rr}}(u, v)}{\sigma} \right) & \widetilde{\mathscr{S}}_{\text{rr}}(u, v) \geq \sigma. \end{cases} \tag{10}$$

For high-rate configurations [quantization parameter (QP) ≤ 22 in HEVC test model (HM) reference software], the reference frames have more complex textures and motions, and the values of $\widetilde{\mathscr{S}}_{\text{rr}}(u, v)$ are increased accordingly. Seven sequences are tested with different QP values to analyze the variation in RQL l, as shown in Table I. As the value of QP decreases, the proposed quantization method achieves a larger l to improve the compression performance.

TABLE I
AVERAGE VALUE OF RQL FOR DIFFERENT QPS

sequence	QP					
	12	17	22	27	32	37
Traffic	0.53	0.43	0.32	0.20	0.09	0.02
crowd_run	0.82	0.78	0.68	0.52	0.32	0.11
PartyScene	0.68	0.56	0.39	0.25	0.12	0.04
BQSquare	0.94	0.84	0.70	0.56	0.38	0.19
Johnny	0.27	0.19	0.11	0.05	0.02	0.00
Aerial	0.94	0.89	0.80	0.68	0.38	0.11
Boat	1.07	1.02	0.94	0.78	0.57	0.32

Using the above-mentioned content-aware adaptive quantization mechanism, the proposed compression algorithm can guarantee the image quality. The RD curve comparisons of the proposed quantization mechanism integrated with the HM15.0 model versus the original HM15.0 model and the fixed length quantization mechanism are shown in Fig. 3. As our algorithm provides almost the same coding efficiency as the standard HM15.0 module, it is hard to distinguish the proposed algorithm's curves from the original ones in most cases. The proposed quantization mechanism can significantly reduce the quality loss compared with the fixed length quantization mechanism for any of the three lengths. The detailed analysis is provided in Section IV.

For the VLSI implementation, the logarithmic and floating-point multiplication operations in (10) have a hardware cost of 49.2 k-gate and only work at 96 MHz in the primitive implementation scheme. In this paper, we propose a lookup table scheme to realize the logarithmic operations in (10).

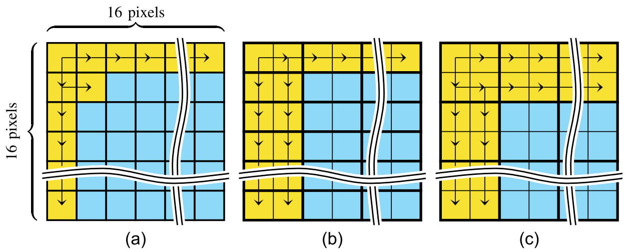


Fig. 4. Pixel locations of parallel prediction. (a) Mode 0 (1 × 1 blocks). (b) Mode 1 (2 × 1 blocks). (c) Mode 2 (2 × 2 blocks).

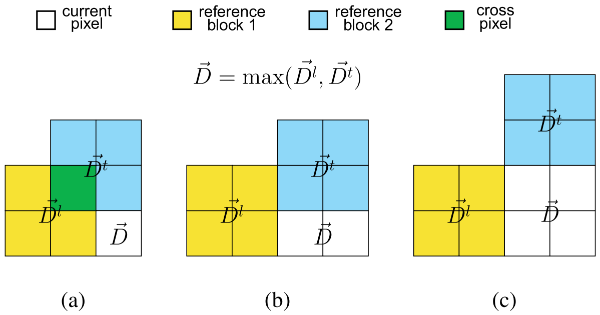


Fig. 5. Calculation of edge vectors for three modes. (a) Mode 0. (b) Mode 1. (c) Mode 2.

To further reduce the hardware cost, in the edge strength calculation [as shown in (7)], we drop the three least significant bits of the input pixels ($p_{i,j}$). The bit truncation scheme can reduce hardware cost by 36.3% and improve the frequency by 15.7%. The detailed implementation and the performance analysis of our RQL decision module are described in Section III.

B. Parallel Directional Prediction

Many methods using intra prediction have been proposed to reduce the energy of coded symbols in FMR algorithms. However, most previous methods merely apply the horizontal and vertical directions, which lower the prediction accuracy. To gain a better prediction performance, additional angular predictions are proposed in the literature [13]. The intra mode is obtained from the HEVC encoder and used to reduce mode decision computation. However, only the coding units (CUs), which have carried out the intra coding, can supply the information to the reference frame encoder. Therefore, the CUs that skip the intra coding will undergo a complex mode decision process.

To overcome the above-mentioned obstacles, this paper proposes a self-contained parallel directional prediction. The prediction directions of the current pixels are estimated by their neighboring left and top pixels. To improve applicability, as shown in Fig. 4, we propose three prediction modes, which can predict one, two, and four pixels in each clock cycle, respectively. For the proposed three prediction modes, throughput is enhanced at the cost of a loss in prediction accuracy. The user can trade off compression efficiency and throughput based on the specific application scenario.

As shown in Fig. 4, the pixels in the yellow region use horizontal or vertical prediction in the direction of the arrow, while those in the sky blue region apply our directional prediction method. As shown in Fig. 5, we estimate the prediction angle of the current block from its left and top 2 × 2-pixel blocks. For the current block, we define the corresponding left and top neighboring edge vectors as $\vec{D}^l = \{dx^l, dy^l\}$ and $\vec{D}^t = \{dx^t, dy^t\}$, respectively, and the reference block with the larger strength value is considered the final reference candidate, i.e., $\vec{D} = \{dx, dy\}$. Some cases (the second row and the second column for Mode 0, and the second row for Mode 1) only have one reference block, so the comparison can be omitted. In $\vec{D}$, dx and dy represent the edge strength in the vertical and horizontal directions [as shown in (7)], respectively. The edge direction of the current pixel can be estimated by the ratio of dy to dx, namely, $\eta(\vec{D}) = dy/dx$. Seven prediction directions are stipulated on the basis of the value of η [24]. To achieve the goal of low-complexity VLSI implementation, the prediction angle (θ) of the current block is obtained by

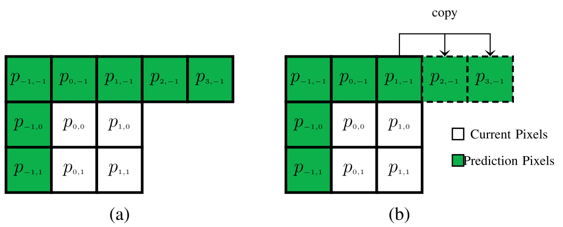


Fig. 6. Reference pixel distribution for the current 2 × 2 pixel block of Mode 2. (a) Normal case. (b) Special case (current block is in the last column).

TABLE II

PREDICT PIXELS OF THE SEVEN CALCULATED DIRECTIONS IN MODE 2

θ	$p'_{0,0}$	$p'_{1,0}$	$p'_{0,1}$	$p'_{1,1}$
45°	$p_{1,-1}$	$p_{2,-1}$	$p_{2,-1}$	$p_{3,-1}$
67.5°	$\frac{p_{0,-1}+p_{1,-1}}{2}$	$\frac{p_{1,-1}+p_{2,-1}}{2}$	$p_{1,-1}$	$p_{2,-1}$
90°	$p_{0,-1}$	$p_{1,-1}$	$p_{0,-1}$	$p_{1,-1}$
112.5°	$\frac{p_{-1,-1}+p_{0,-1}}{2}$	$\frac{p_{0,-1}+p_{1,-1}}{2}$	$p_{-1,-1}$	$p_{0,-1}$
135°	$p_{-1,-1}$	$p_{0,-1}$	$p_{-1,0}$	$p_{-1,-1}$
157.5°	$\frac{p_{-1,-1}+p_{-1,0}}{2}$	$p_{-1,-1}$	$\frac{p_{-1,0}+p_{-1,1}}{2}$	$p_{-1,0}$
180°	$p_{-1,0}$	$p_{-1,0}$	$p_{-1,1}$	$p_{-1,1}$

$$\theta = \begin{cases} 45^\circ & \text{if } 0.5 < \eta(\vec{D}) \leq 2 \\ 67.5^\circ & \text{if } 2 < \eta(\vec{D}) \leq 4 \\ 90^\circ & \text{if } |\eta(\vec{D})| > 4 \\ 112.5^\circ & \text{if } -4 \leq \eta(\vec{D}) \leq -2 \\ 135^\circ & \text{if } -2 < \eta(\vec{D}) \leq -1 \\ 157.5^\circ & \text{if } -1 < \eta(\vec{D}) \leq -0.25 \\ 180^\circ & \text{if } -0.25 < \eta(\vec{D}) \leq 0.5. \end{cases} \tag{11}$$

The reference pixel distribution for the current 2 × 2 pixel block of Mode 2 is shown in Fig. 6. For a block in the last column, as the reference pixels on the top-right do not exist, $p_{1,-1}$ is copied to fill the two blank positions. The prediction pixels for the seven directions calculated in Mode 2 are shown in Table II. If the prediction angle points at the middle of two pixels, we calculate the prediction pixel by averaging the two pixels; otherwise, we use the pixel located in the prediction direction. The cases for Mode 0 and Mode 1 can be determined accordingly.

TABLE III
PREDICTION PERFORMANCE ANALYSIS IN TERMS OF SUM OF SQUARED ERROR FOR ONE 16 × 16 LUMA BLOCK (QP = 32)

sequence	original$^{1)}$	M0$^{2)}$	M1$^{3)}$	M2$^{4)}$
PeopleOnStreet	21247	9201	9749	11040
Traffic	11136	5434	5774	6450
ParkScene	7924	6339	6688	7149
Tennis	4696	1469	1575	1726
BasketBallDrill	24522	10221	10818	12095
RaceHorses	30362	21762	21997	23119
BasketballPass	28921	13222	13590	14402
BlowingBubbles	43484	19852	20473	20986
Johnny	12985	3123	3265	3530
KristenAndSara	26790	6686	6975	7277
average	21206	9730	10090	10777

$^{1)}$ original horizontal/vertical prediction
$^{2)}$ proposed Mode 0 prediction
$^{3)}$ proposed Mode 1 prediction
$^{4)}$ proposed Mode 2 prediction

TABLE IV
SUM OF SQUARED ERROR OF THE PREDICTION RESIDUES FOR DIFFERENT QPs. (MODE = 0)

sequence	QP					
	12	17	22	27	32	37
Traffic	7566	7132	6644	6071	5434	4725
crowd_run	40628	39908	37093	33765	29086	23383
PartyScene	23308	22241	20562	18963	16785	13416
BQSquare	114029	112639	108317	100941	92133	80727
Johnny	4357	3947	3655	3390	3123	2825
Aerial	116776	108906	97903	83881	66061	42652
Boat	72417	64058	56852	52575	48042	41599

Ten typical sequences (200 frames for each sequence, QP = 32) are tested and the prediction performance analysis in terms of sum of squared error for one 16 × 16 luma block is shown in Table III. When compared with the original vertical/horizontal prediction, Mode 0, Mode 1, and Mode 2 prediction schemes can reduce the error energy by 54.1%, 52.4%, and 49.2%, respectively.

There are complex textures in the reference frame for high-rate configurations, and the amplitudes of the prediction residues increase accordingly. The sum of the squared error of the prediction residues for different QP values is shown in Table IV. The experiments reveal that the energy of the prediction residues increases noticeably for a low value of QP, which will decrease the compression efficiency of FMR algorithms; however, the proposed content-aware adaptive quantization mechanism can mitigate this effect and achieve a stable compression performance. The detailed analysis is described in Section IV.

C. Dynamic kth-Order Unary/Exp-Golomb Coding

The adaptive order unary/Exp-Golomb coding is used as our entropy coding algorithm. In particular, when given the order k, for the input value x, we have the quotient $q = x/2^k$ and the remainder $\omega = x\%2^k$. As the Exp-Golomb coding has a poor compression performance when coding small residues, we introduce the unary coding [25] to compress small values. However, the basic Exp-Golomb coding [26] cannot combine seamlessly with the unary coding. Therefore, for the quotient part, we use the improved unary/Exp-Golomb coding with a threshold of 4. That is, when $q < 4$, we use the unary coding; otherwise, the improved Exp-Golomb coding is applied. The remainder, ω, is transmitted following the coded quotient. Examples with the values of order $k \in \{0, 1, 2, 3\}$ are illustrated in Table V. The underscore separates the quotient part and the remainder part. If $x \neq 0$, the sign bit needs to be transmitted after the remainder part.

TABLE V
UNARY/Exp-Golomb CODING WITH ORDER $k \in \{0, 1, 2, 3\}$

Coding	order k			
residue	k=0	k=1	k=2	k=3
0	0_	0_0	0_00	0_000
±1	10_S	0_1S	0_01S	0_001S
±2	110_S	10_0S	0_10S	0_010S
±3	1110_S	10_1S	0_11S	0_011S
±4	111100_S	110_0S	10_00S	0_100S
±5	111101_S	110_1S	10_01S	0_101S
±6	11111000_S	1110_0S	10_10S	0_110S
±7	11111001_S	1110_1S	10_11S	0_111S
±8	11111010_S	111100_0S	110_00S	10_000S
...	...	...	...	...
±15	...	...	1110_11S	10_111S
±16	...	...	111100_00S	110_000S
...	...	...	...	...
±31	...	...	...	1110_111S
±32	...	...	...	111100_000S
...	...	...	...	...

In a manner similar to the pixel prediction process, we apply the parallel directional prediction process to update the k-order. Through simplification, we define four directions, and the order $k_{i,j}$ of the current pixel is deduced from previously coded pixels. The relationship between the prediction angle and the order k in Mode 2 is shown in Table VI. The prediction of order k in Mode 0 and Mode 1 can be deduced accordingly. It should be noted that $k'_{i,j}$ is a fine-tuned value of the adopted order $k_{i,j}$. As described by (12), if the decoded residue value $x_{i,j}$ is not in the optimal expression range, i.e., $[2^{k_{i,j}-1}, 3\times 2^{k_{i,j}}]$, we will adjust $k_{i,j}$ by −1 or +1 accordingly. The value of $k_{0,0}$ is defined as 1, which achieves a 0.2% DRR increase compared with other counterparts [16]

$$k'_{i,j} = \begin{cases} k_{i,j} + 1 & x \geq 3 \times 2^{k_{i,j}} \, (k_{i,j} < 3) \\ k_{i,j} & 2^{k_{i,j}-1} \leq x < 3 \times 2^{k_{i,j}} \\ k_{i,j} - 1 & x < 2^{k_{i,j}-1} (k_{i,j} > 0). \end{cases} \tag{12}$$

Many all-zero-residue regions are found in the chroma partitions, and this provides great potential for further improving the compression ratio. Based on this property, two compression skip flags (CSFs), i.e., the partition CSF (PCSF) and

TABLE VI

RELATIONSHIP BETWEEN THE PREDICTION ANGLE AND THE ORDER k IN MODE 2

θ	$k_{i,j}$	$k_{i+1,j}$	$k_{i,j+1}$	$k_{i+1,j+1}$
45°	$k'_{i+1,j-1}$	$k'_{i+2,j-1}$	$k'_{i+2,j-1}$	$k'_{i+3,j-1}$
90°	$k'_{i,j-1}$	$k'_{i+1,j-1}$	$k'_{i,j-1}$	$k'_{i+1,j-1}$
135°	$k'_{i-1,j-1}$	$k'_{i,j-1}$	$k'_{i-1,j}$	$k'_{i-1,j-1}$
180°	$k'_{i-1,j}$	$k'_{i-1,j}$	$k'_{i-1,j+1}$	$k'_{i-1,j+1}$

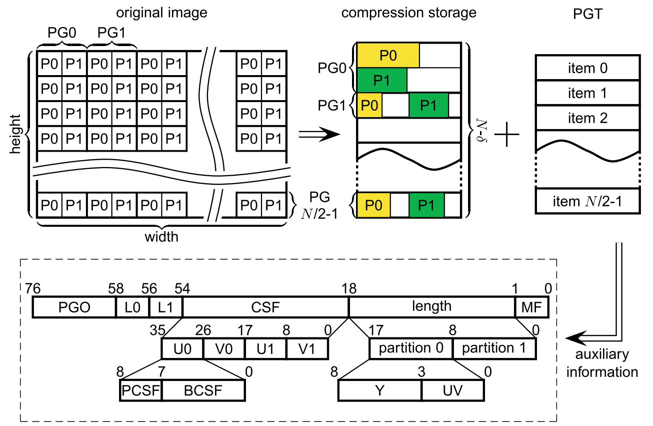


Fig. 7. Memory mapping and auxiliary information for the compression partition. N = height × width/16/16. δ: number of PGs that can be stored in the compression mode. PG offset: record the beginning address of the PG. L0 and L1: RQLs of the two partitions. CSF: PCSF and BCSF of the chroma components in a PG. Length: length of a compressed component in the units of the IO bit width. Merge flag: indicate whether to use the compression storage scheme.

block CSF (BCSF), are proposed in this paper. When the PCSF is set, the 8 × 8 chroma partition is an all-zero-residue partition. If the residues in one 4 × 2 block are all zeros, the associated 1-b BCSF is set. By applying the CSF method, the DRR of the chroma partitions can be increased an average of 6.0% [16].

D. Partition Group Table-Based Compression Storage

Although the above-mentioned methods can achieve an average DRR of over 70%, they still cannot contribute to the memory storage optimization. Because of the varying DRR of our algorithm, the compressed partitions cannot be addressed linearly. A PG table-based storage method is proposed in this paper to resolve the address mapping issues.

Every two horizontally adjacent 16 × 16 partitions compose one PG. One dedicated PG table is assigned to each PG, containing the auxiliary information of the two partitions used for decompression. The memory storage mapping and the auxiliary information are shown in Fig. 7. When the DRRs of the two partitions in one PG are all no less than 50%, the content of one PG can be stored in the space of one partition; otherwise, space for two partitions is still needed. To allow the decompressor to reduce the number of unnecessary data read operations and to determine the optimal BL, the PG table also provides length information. The original data are stored when the DRR is less than 0. When the length value is equal to the original length, the decompressor skips the decompression process.

By applying the proposed storage scheme, our algorithm can save external DRAM memory space and reduce the frequency of the precharge and activate operations accordingly. Over 38% of the dynamic power of DRAM is consumed by the precharge and activate operations [6]. Therefore, the proposed storage scheme is an efficient method of reducing the power requirements of DRAM. The detailed analysis of the external DRAM dynamic power is described in Section IV.

III. VLSI IMPLEMENTATION

Based on the aforementioned FMR algorithm, we develop the associated VLSI implementation. Fig. 8 shows the overall architecture of the proposed reference frame compressor/decompressor.

A. VLSI Design of the Compressor

The reconstructed pictures from the HEVC encoder/decoder are divided into 16 × 16 pixel partitions and are transmitted to our compressor. To increase the clock speed, the encoder engine is composed of three pipeline stages: 1) directional prediction; 2) residue quantization; and 3) entropy encoding. As the quantized residues are needed by the directional predictor to predict the direction of the next pixel block, data dependence will seriously degrade the pipeline utilization of the primitive scheduling method, as shown in Fig. 9(a). In the primitive design, only one block is coded in every two cycles, degrading the hardware utilization to 50%. In this paper, we apply the interchange compression of Y odd block row and Y even block row components, described in Fig. 9(b), and U- and V-components, described in Fig. 9(c), to increase the pipeline efficiency. With the proposed scheduling mechanism, 100% hardware utilization can be achieved. The wave-front mode is adopted in this paper to alleviate the dependence between the Y odd block row and Y even block row components. The compression process of the Y odd block row should be at least two blocks in advance of that of the Y even block row.

The values of the previous k' should be buffered to obtain the k value of the current pixel. In our design, by using the wave-front mode, we can discard the k' values of the Y odd block row that are no longer required by the Y even block row, and the freed space can store the new k' value of the Y even block row. Consequently, the buffer size is reduced to $(16+2)\times 2 = 36$ b.

For a fixed QP, the RQL l increases with $\widetilde{\mathscr{F}}_{\rm rr}(u,v)$, as shown in Fig. 10. As l is an integer value, we just need to know the thresholds of $\widetilde{\mathscr{F}}_{\rm rr}(u,v)$. In this paper, a lookup table scheme is adopted to realize the logarithmic operations. Based on the principle of rounding, we choose the values of $\widetilde{\mathscr{F}}_{\rm rr}(u,v)$ when $l = 0.5$, $l = 1.5$, and $l = 2.5$ ($\Gamma_{0,1}$, $\Gamma_{1,2}$, and $\Gamma_{2,3}$ in Fig. 10) as the thresholds. We can observe from Fig. 10 that the thresholds increase with high QP, and the thresholds for QP $= 22 - 37$ are shown in Table VII. The bit truncation scheme is also adopted in this paper to further reduce the hardware cost. The performance analysis of the bit truncation scheme is shown in Table VIII. The experiments show that the

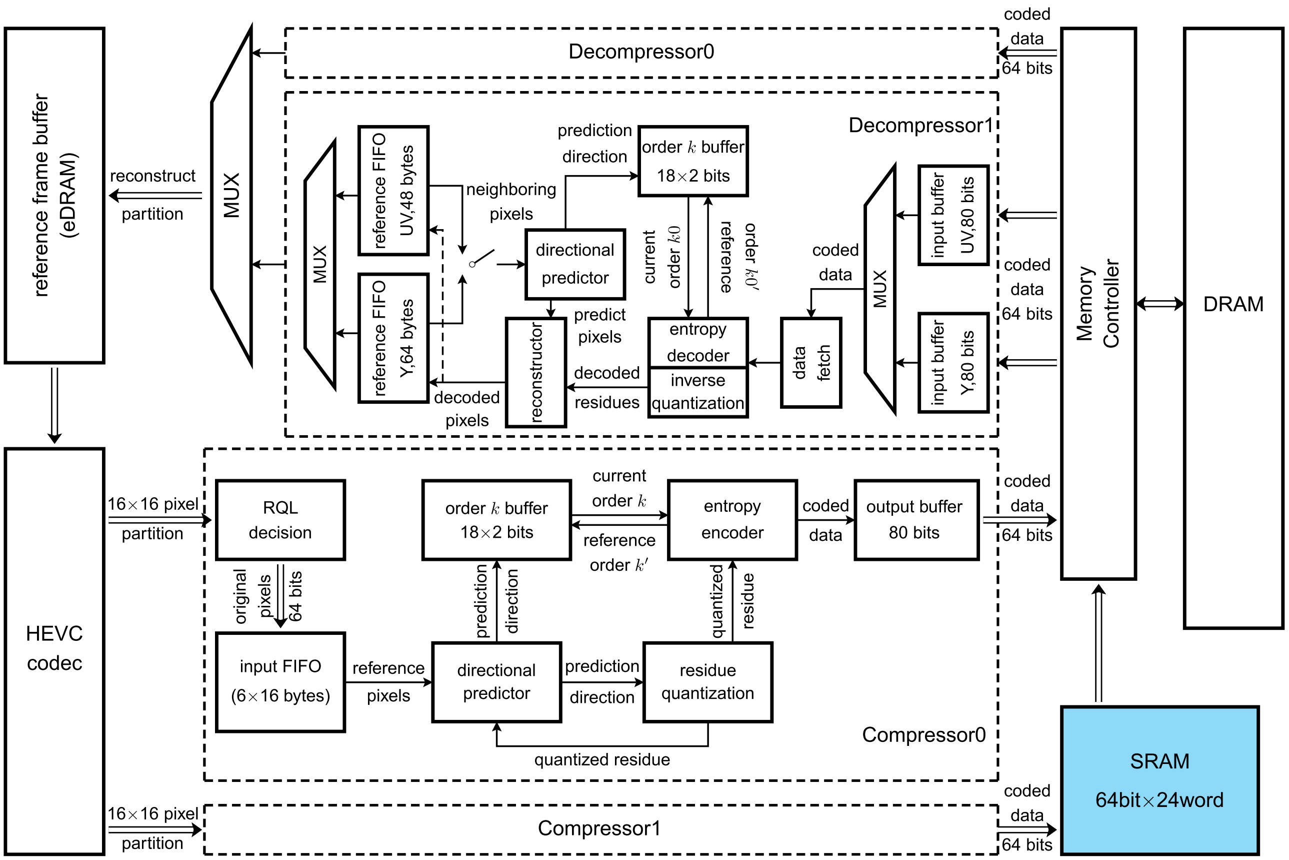


Fig. 8. Overall architecture of the proposed frame memory compressor/decompressor (the parameters in the figure are for Mode 0).

TABLE VII

THRESHOLD OF $\mathscr{T}_{\Gamma\Gamma}(u, v)$ IN LOOKUP TABLE ($\gamma = 0.0002$)

QP	22	23	24	25	26	27	28	29
$\Gamma_{0,1}$	19	20	21	22	23	24	25	26
$\Gamma_{1,2}$	65	67	69	71	73	76	79	82
$\Gamma_{2,3}$	219	225	231	238	245	252	260	268
QP	30	31	32	33	34	35	36	37
$\Gamma_{0,1}$	27	28	30	31	33	35	37	39
$\Gamma_{1,2}$	85	88	92	96	100	104	109	114
$\Gamma_{2,3}$	276	285	295	305	316	328	340	353

bit truncation scheme achieves a decrease in an area of 36.3% and an increase in a frequency of 15.7% when compared with the primitive scheme for the RQL decision module, while the average DRR degradation introduced by the bit truncation is only 0.24%.

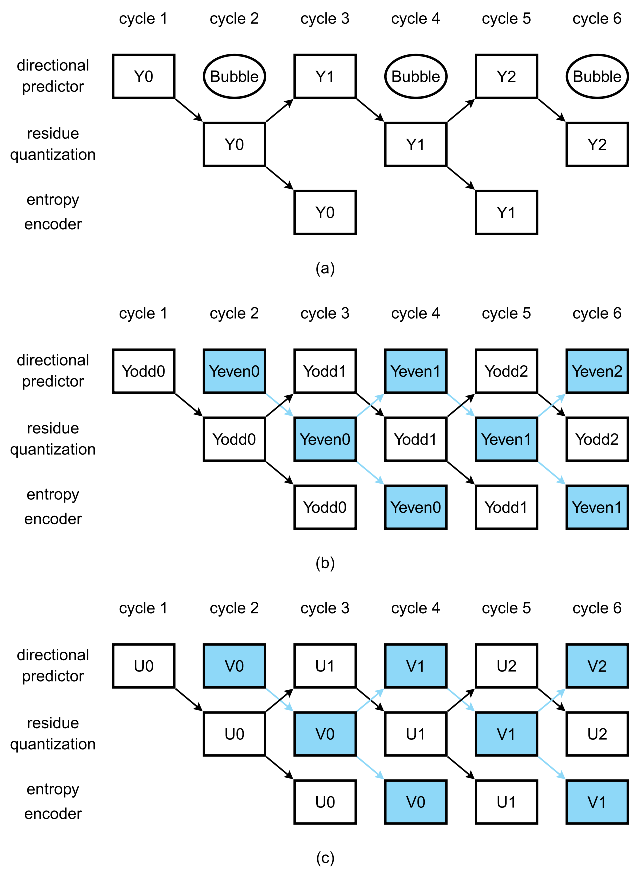
TABLE VIII

PERFORMANCE ANALYSIS OF THE BIT TRUNCATION SCHEME. (QP = 22, 27, 32, 37, MODE = 0, AND $\gamma = 0.0002$)

sequence	no truncation		truncation	
	BD-PSNR(dB)	DRR(%)	BD-PSNR(dB)	DRR(%)
PeopleOnStreet	-0.0272	70.98	-0.0222	70.60
Traffic	-0.0773	70.95	-0.0737	70.67
ParkScene	-0.0547	70.61	-0.0496	71.34
Tennis	-0.0232	78.21	-0.0198	77.91
BasketBallDrill	-0.0479	67.62	-0.0482	67.62
RaceHorses	-0.0240	64.36	-0.0224	64.10
BasketballPass	-0.0236	70.44	-0.0163	70.02
BlowingBubbles	-0.0495	59.86	-0.0424	59.58
Johnny	-0.0517	81.39	-0.0539	81.42
KristenAndSara	-0.0343	80.32	-0.0325	80.10
average	-0.0413	71.58	-0.0381	71.34
Gate count(K)	3.876		2.468	
Frequency(MHz)	520.8		602.4	

B. VLSI Design of the Decompressor

The decompressor is composed of four pipeline stages, including: 1) data fetch; 2) entropy decoding and inverse quantization; 3) reconstruction; and 4) directional prediction. To shorten the critical path, the reconstructed pixels are first written back to the register and then used for directional prediction in the next pipeline stage, as shown in Fig. 11(a). As the order k value in the data fetch stage of the current block (Y_n) depends on the directional prediction result of the previous block (Y_{n-1}), the primitive schedule will lead to bubbles in pipelining and degrade hardware utilization. In every four cycles, only one block is decoded, and hardware utilization is only 25%. To improve the pipeline efficiency, the interchange decoding of Y odd block row, Y even block row, and U- and V-components is adopted in this paper,



	cycle 1	cycle 2	cycle 3	cycle 4	cycle 5	cycle 6
directional predictor	Y0	Bubble	Y1	Bubble	Y2	Bubble
residue quantization		Y0		Y1		Y2
entropy encoder			Y0		Y1	

(a)

	cycle 1	cycle 2	cycle 3	cycle 4	cycle 5	cycle 6
directional predictor	Yodd0	Yeven0	Yodd1	Yeven1	Yodd2	Yeven2
residue quantization		Yodd0	Yeven0	Yodd1	Yeven1	Yodd2
entropy encoder			Yodd0	Yeven0	Yodd1	Yeven1

(b)

	cycle 1	cycle 2	cycle 3	cycle 4	cycle 5	cycle 6
directional predictor	U0	V0	U1	V1	U2	V2
residue quantization		U0	V0	U1	V1	U2
entropy encoder			U0	V0	U1	V1

(c)

Fig. 9. Processing schedule of (a) primitive compression architecture, (b) proposed interchange architecture for luma components, and (c) proposed interchange architecture for chroma components. Y0, Y1, and Y2: blocks of Y-components. Yodd0, Yodd1, and Yodd2: blocks in Y odd block row. Yeven0, Yeven1, and Yeven2: blocks in Y even block row. U0, U1, U2, V0, V1, and V2: blocks of U- and V-components.

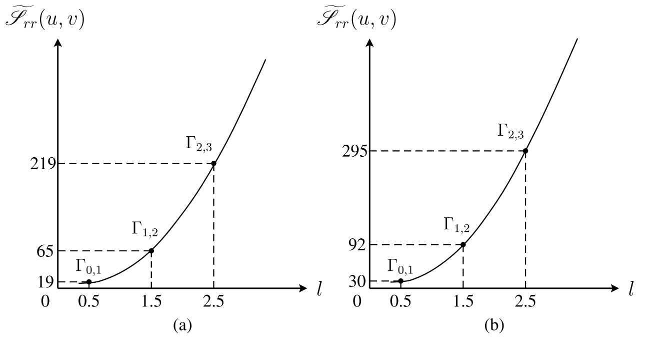
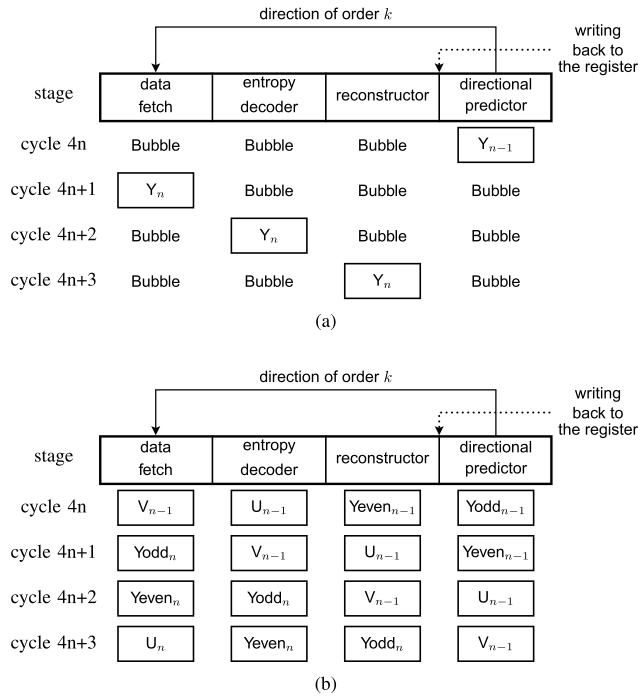


Fig. 10. Relationship between l and $\widetilde{\mathscr{F}}_{\mathrm{rr}}(u,v)$. (a) QP = 22. (b) QP = 32.



stage	data fetch	entropy decoder	reconstructor	directional predictor
cycle 4n	Bubble	Bubble	Bubble	Y_{n-1}
cycle 4n+1	Y_n	Bubble	Bubble	Bubble
cycle 4n+2	Bubble	Y_n	Bubble	Bubble
cycle 4n+3	Bubble	Bubble	Y_n	Bubble

(a)

stage	data fetch	entropy decoder	reconstructor	directional predictor
cycle 4n	V_{n-1}	U_{n-1}	Yeven_{n-1}	Yodd_{n-1}
cycle 4n+1	Yodd_n	V_{n-1}	U_{n-1}	Yeven_{n-1}
cycle 4n+2	Yeven_n	Yodd_n	V_{n-1}	U_{n-1}
cycle 4n+3	U_n	Yeven_n	Yodd_n	V_{n-1}

(b)

Fig. 11. Pipeline optimization of the decompressor. (a) Primitive architecture. (b) Proposed interchange architecture.

as described by Fig. 11(b). By adopting the interchange schedule, hardware utilization is enhanced to 100%.

To further improve the throughput, a two-parallel-compressor (decompressor) structure is adopted in this paper, which can simultaneously handle the two partitions in one PG, as shown in Fig. 8. During the encoding, as the start address of the first partition in the PG has been determined, the PG's coded data can be directly dispatched in 64-b granularity to the memory controller [27]. However, the start address of the second partition is unknown, because we do not know whether the compressed storage scheme could be applied at the initial stage of the compression. In this paper, the coded data of the second partition are temporally cached in 64 b × 24 word SRAM (the blue block in Fig. 8), which is half of the original partition size. As described earlier, if either of the two partitions achieves a less than 50% DRR, the second partition will be stored in the next partition space. Otherwise, the two partitions can share one partition space. That is, if overflow occurs when buffering partition 2's output stream, the start address of partition 2 is assigned to a separate space; otherwise, not until the DRRs of partition 1 and partition 2 are known is the storage scheme decided.

IV. Experimental Results

In this section, we first evaluate the compression performance and DRAM power reduction of the proposed lossy FMR algorithm. Thereafter, the throughput, the hardware cost, and the power dissipation of our hardwired reference frame codec are explained. Finally, comparisons of the proposed architecture with previous designs are described in detail.

A. Compression Performance Analysis

The proposed method is conducted on HEVC reference test model HM15.0 [28]. In this experiment, 26 typical video sequences in classes A–F, seven HD sequences, and two 4K sequences are tested to analyze compression performance.

The variable γ is an important factor affecting the quantization result. Four sets of experiments with different γ values are run to analyze the compression performance of content-aware adaptive quantization, as shown in Table IX. The coding

TABLE IX

Compression Performance of Content-Aware Adaptive Quantization (QP = 22, 27, 32, 37 and Mode = 0)

sequence	BD-PSNR(dB)				BD-BR(%)				DRR(%)			
	γ				γ				γ			
	0.00002	0.0002	0.002	0.02	0.00002	0.0002	0.002	0.02	0.00002	0.0002	0.002	0.02
PeopleOnStreet	-0.0229	-0.0222	-0.0660	-0.2703	+0.57	+0.56	+1.67	+7.14	70.45	70.60	70.75	72.62
Traffic	-0.0749	-0.0737	-0.1275	-0.5504	+2.69	+2.63	+4.63	+22.63	70.62	70.67	70.80	72.09
ParkScene	-0.0477	-0.0496	-0.0730	-0.3500	+1.73	+1.80	+2.69	+14.54	71.33	71.34	71.52	73.16
Tennis	-0.0184	-0.0198	-0.0271	-0.1345	+0.68	+0.74	+1.01	+5.21	77.93	77.91	77.98	78.47
BasketBallDrill	-0.0453	-0.0482	-0.0886	-0.3388	+1.25	+1.34	+2.50	+10.34	67.56	67.62	67.77	68.75
RaceHorses	-0.0235	-0.0224	-0.1141	-0.3772	+0.65	+0.64	+3.28	+11.29	63.79	64.10	65.31	68.55
BasketballPass	-0.0177	-0.0163	-0.0618	-0.2379	+0.40	+0.37	+1.44	+5.90	69.94	70.02	70.39	71.28
BlowingBubbles	-0.0447	-0.0424	-0.1178	-0.4710	+1.30	+1.24	+3.52	+15.72	59.42	59.58	60.12	62.81
Johnny	-0.0507	-0.0539	-0.0678	-0.2626	+2.12	+2.14	+2.94	+12.50	81.42	81.42	87.47	81.72
KristenAndSara	-0.0332	-0.0325	-0.0475	-0.2336	+1.22	+1.24	+1.75	+9.00	80.07	80.10	80.12	80.35
average	-0.0379	-0.0381	-0.0791	-0.3226	+1.26	+1.27	+2.54	+11.43	71.25	71.34	71.62	72.98

TABLE X

Compression Performance Comparison of the Proposed Mode 0 and Fixed Length Quantization Mechanism (γ = 0.0002)

	QP	Mode 0	fixed quantization length		
			1	2	3
DRR(%)	12	63.89	66.97	77.54	83.83
	17	66.90	70.19	78.67	84.41
	22	69.18	72.86	80.08	85.23
	27	70.11	74.70	81.42	86.03
	32	71.03	76.21	82.29	86.62
	37	72.13	77.67	83.24	87.21
	average	68.87	73.10	80.54	85.56
BD-PSNR(dB)		-0.0442	-0.5303	-1.6875	-3.4071

TABLE XI

DRR of Three Prediction Modes (QP = 22, 27, 32, 37 and γ = 0.0002)

class	video sequence	Mode 0(%)	Mode 1(%)	Mode 2(%)
A	*PeopleOnStreet*	70.60	69.09	64.98
	Traffic	70.67	69.43	65.85
B	*crowd_run*	61.36	59.27	55.26
	BasketballDrive	78.69	78.33	73.14
	BQTerrace	70.86	69.31	65.12
	Cactus	71.27	70.07	65.92
	Kimono1	74.59	74.04	69.85
	ParkScene	71.34	70.12	66.36
	Tennis	77.91	77.46	72.46
C	*BasketballDrill*	67.62	66.14	62.99
	BQMall	67.29	65.56	61.70
	PartyScene	55.60	52.88	48.75
	RaceHorses	64.10	62.41	57.96
D	*BasketballPass*	70.02	68.71	65.12
	BlowingBubbles	59.58	57.64	55.02
	BQSquare	62.05	58.75	54.33
	RaceHorses	61.11	59.15	55.27
E	*Johnny*	81.42	80.82	75.25
	KristenAndSara	80.10	79.36	73.74
	vidyo1	79.37	78.63	73.68
	vidyo3	80.70	80.06	74.22
	vidyo4	80.34	79.73	76.61
F	*BasketballDrillText*	67.66	66.21	62.59
	SlideEditing	70.43	68.84	63.45
average		70.61	69.25	64.90

quality is measured by Bjøntegaard delta peak signal-to-noise rate (BD-PSNR) and Bjøntegaard delta bit rate (BD-BR). As the value of γ increases, the DRR of the reference frame is increased, while the coding quality is degraded. For instance, when $\gamma = 0.0002$, the average DRR is 71.34% with BD-BR = +1.27%. In contrast, when γ is increased to 0.02, the DRR grows to 72.98% at the cost of a BD-BR = +11.43%. By adjusting the parameter γ, we can make a tradeoff between compression efficiency and image quality.

After defining the parameter γ using the experiments mentioned earlier, the content-aware adaptive quantization mechanism has been determined. The mechanism is able to further improve the compression efficiency of the lossy FMR algorithm while maintaining the overall encoding quality. The compression performance comparison of the proposed quantization mechanism and the fixed length quantization mechanism is shown in Table X. The fixed length quantization mechanism can obviously improve the DRR; however, it introduces unacceptable quality loss. The proposed quantization mechanism reduces the quality loss by 0.49, 1.64, and 3.36 dB in terms of BD-PSNR compared with the fixed length quantization mechanism with three different respective quantization lengths.

To support a variety of application scenarios, we propose three prediction modes, which can process one, two, and four pixels in each cycle, respectively. The compression performance figures are shown in Table XI. Mode 0 achieves the highest DRR (70.61%), but can only process one pixel per cycle. Mode 1 can double the throughput of Mode 0 with a 1.36% DRR decrease. Mode 2 quadruples the throughput of Mode 0 at the cost of a 5.71% DRR decrease.

The compression performance comparisons of the proposed algorithm and two other lossy algorithms in terms of DRR are shown in Table XII. The proposed algorithm can achieve

TABLE XII

Compression Performance Comparison of the Proposed Mode 0 and Other Lossy Algorithms ($\gamma = 0.0002$)

video sequence	QP	Mode 0	Tsai's [8]	Fan's [13]
Bluesky	15	77.37%	60.32%	65.12%
	20	78.38%	67.11%	66.46%
	25	78.19%	68.25%	66.85%
Rushhour	15	78.08%	58.51%	61.75%
	20	80.18%	68.25%	69.02%
	25	80.40%	71.75%	71.01%
Station2	15	74.72%	53.27%	64.30%
	20	75.14%	58.51%	65.80%
	25	75.80%	62.83%	66.87%
Sunflower	15	73.76%	61.24%	63.37%
	20	74.42%	65.16%	68.45%
	25	74.45%	66.22%	68.77%

TABLE XIII

DRR of the Proposed Mode 0, Guo *et al.*'s [11] and Lian *et al.*'s [16] Algorithms for Different QPs (Luma Only and $\gamma = 0.0002$)

video sequence	QP	Mode 0	Lian's [16]	Guo's [11]
Traffic	12	62.55%	54.79%	51.37%
	17	64.01%	58.01%	54.10%
	22	64.31%	59.91%	56.25%
	27	64.26%	61.43%	57.81%
	32	64.16%	62.84%	59.57%
	37	64.60%	64.21%	61.13%
crowd_run	12	47.17%	35.69%	30.47%
	17	48.88%	37.94%	32.81%
	22	53.27%	43.95%	38.67%
	27	54.59%	47.46%	42.58%
	32	55.08%	50.68%	45.90%
	37	55.22%	53.81%	49.41%
PartyScene	12	44.34%	31.54%	22.85%
	17	45.70%	33.11%	24.41%
	22	46.92%	35.55%	27.15%
	27	48.00%	38.96%	31.05%
	32	47.41%	42.29%	34.96%
	37	47.22%	45.80%	39.65%
BQSquare	12	48.54%	34.42%	25.20%
	17	49.07%	36.28%	27.15%
	22	48.88%	38.33%	29.49%
	27	50.34%	42.24%	33.98%
	32	54.39%	49.12%	41.99%
	37	53.47%	50.93%	44.73%
Johnny	12	67.19%	63.04%	60.74%
	17	72.85%	70.12%	67.97%
	22	75.10%	73.58%	72.66%
	27	75.15%	74.41%	73.83%
	32	75.34%	75.10%	74.61%
	37	76.17%	76.12%	75.78%
Aerial	12	52.89%	43.20%	34.38%
	17	53.16%	43.98%	35.16%
	22	53.32%	44.96%	36.41%
	27	53.40%	46.13%	38.28%
	32	51.88%	48.20%	41.25%
	37	52.70%	51.91%	46.09%
Boat	12	59.34%	48.13%	40.63%
	17	60.23%	49.69%	42.66%
	22	60.51%	50.74%	44.06%
	27	59.65%	51.33%	45.00%
	32	57.85%	52.03%	45.94%
	37	56.48%	53.48%	47.81%

a 5.7%–21.5% DRR increase when compared with the other algorithms.

Through the analysis of Tables I and IV, the amplitudes of the prediction residues in the reference frames can be seen to increase significantly for high-rate configurations, which will decrease the compression efficiency of lossless algorithms. The proposed lossy algorithm can mitigate the effect and obtain a stable compression performance by increasing the RQL l. Five typical and two 4K sequences are tested. Table XIII illustrates the compression comparisons of the proposed Mode 0, Guo *et al.*'s [11] and Lian *et al.*'s [16] algorithms for different QPs. Taking as an example the *crowd_run* sequence, the proposed algorithm can achieve the DRR improvements of 1.41%–5.81% compared with the previous lossless algorithms when QP $=$ 37. When the QP is decreased to 12, the improvements grow to 11.48%–16.7%.

The HEVC profile Main15 supports videos with bit depths beyond 8 b per sample ($>$24 bit per pixel), and one 10-b sample (Y, Cb, or Cr) needs a 16-b storage space. Therefore, the bit-depth extension wastes a large amount of memory space and increases the bandwidth requirements. For sequences over 24 bit per pixel, if the average bit depth of one partition can be compressed to 8 b, one 16×16 partition in our algorithm can be stored in a $(16 \times 16 + 8 \times 8 \times 2) \times 8$-b region, instead of occupying a $(16 \times 16 + 8 \times 8 \times 2) \times 16$-b area. The bandwidth and memory requirements of the DRAM can be significantly reduced. Our experimental results in Table XIV reveal that 87.83%–96.73% of the partitions in 30-bit per pixel video sequences can be compressed to the average 8-b space, 4.76%–14.2% more than in our previous lossless algorithm [16], with a quality loss of only -0.02 dB in terms of BD-PSNR, and 19.5%–61.9% more than Guo *et al.*'s [11] algorithm. On average, 94.46% of the partitions can be saved in half of a partition space. Therefore, the number of DRAM activate and precharge operations can be reduced to 47.2%.

Our reference frame recompression algorithm also contributes to a reduction in DRAM dynamic power consumption, and the CACTI simulator [29] is introduced to evaluate the dynamic power usage of the DRAM with 1.5 V core and IO voltages. The DRAM core structure is composed of multiple identical banks that can be accessed simultaneously. The bank is organized as a 2D array consisting of rows and columns. The read and write accesses to the DRAM are burst-oriented. The activate operation is used to select the bank and row to be accessed and the data in the selected row are transmitted to the row buffer. Then, the read and write operations select the starting column address for the burst access. When the

TABLE XIV

COMPARISONS OF THE PROPOSED MODE 0, GUO *et al.*'s [11] AND LIAN *et al.*'s [16] ALGORITHMS IN THE COMPRESSION OF 30-BIT PER PIXEL SEQUENCES (THE METRIC IS THE RATIO OF 16 × 16 PARTITIONS THAT CAN BE COMPRESSED IN (16 × 16 + 8 × 8 × 2) × 8-b MEMORY REGIONS, $\gamma = 0.0002$)

video sequence	QP	Mode 0	Lian's [16]	Guo's [11]
NebutaFestival	22	87.83%	74.47%	25.97%
	27	91.92%	77.72%	34.37%
	32	95.70%	84.65%	58.11%
	37	96.45%	88.71%	69.98%
SteamLocomotiveTrain	22	95.64%	87.86%	68.90%
	27	95.56%	89.02%	71.07%
	32	95.86%	90.12%	73.44%
	37	96.73%	91.97%	77.20%

TABLE XV

VLSI IMPLEMENTATION RESULTS FOR THE COMPRESSORS AND DECOMPRESSORS OF THE THREE MODES. (VOLTAGE = 0.9 V AND TEMPERATURE = 125 °C)

	Mode 0		Mode 1		Mode 2	
	Com.	Dec.	Com.	Dec.	Com.	Dec.
CMOS technology	65nm					
Gate count(K)	24.6	29.5	33.6	38.6	51.4	59.3
On-chip SRAM(byte)	0					
Max freq.(MHz)	562	578	546	565	543	424
Power consumption(mW)	3.6	4.4	5.3	5.8	9.6	11.0

desired data are not located in the activated row, the precharge operation is used to save the current row data back to the memory bank and close the activated bank and row. The power consumed by the precharge and activate operations accounts for 38.2% of the total dynamic power consumption of the DRAM [30]. On average, our memory storage scheme decreases the frequency of precharge and activate operations by 41%. Accordingly, our experiments show that the proposed method achieves DRAM dynamic power consumption savings of 15.7%.

B. Hardware Implementation Analysis

The hardware implementation results are presented in this section. Using TSMC 65-nm CMOS technology, the design has been described using Verilog-HDL and was synthesized by the Synopsys Design Compiler. IC-Compiler implements the place and route jobs.

Table XV shows the VLSI implementation results for the three modes' compressors and decompressors. In the worst working conditions (0.9 V and 125 °C), the compressor achieves a 562-MHz clock speed with only 24.6-k-gate standard cells, while the decompressor consumes 29.5-k-gate standard cells at a frequency of 578 MHz in Mode 0. For Mode 1 and Mode 2, the hardware costs of the compressor (decompressor) are increased to 33.6 k-gate (38.6 k-gate) and 51.4 k-gate (59.3 k-gate), respectively. For the decompressor of Mode 2, the data fetch of four pixels in one block is a serial process, extending the critical path. Therefore, the maximum frequency of the decompressor is decreased to 424 MHz in Mode 2. The compressor and decompressor consume 3.6 mW (5.3 and 9.6 mW) and 4.4 mW (5.8 and 11.0 mW) of power, respectively.

Comparisons of the hardware implementation of the proposed lossy architecture with those of previous lossless and lossy architectures are shown in Table XVI. Lee *et al.*'s [12] work can obtain a fixed DRR of 50% and halves the memory storage requirements. The indistinctive lossy compression to the reference frame pixel in [12] results in the serious coding quality loss (ΔPSNR $= -1.03$ dB). By applying the discrete wavelet transform and set partitioning in hierarchical trees (SPIHT) methods, a DRR of 65.3% with a −0.10-dB image quality change is achieved in [14]. Because of its complex transform and SPIHT coding, its clock speed is only 10 MHz and the throughput is as low as 4.5 Mpixels/s, which can support only a common intermediate format (352 × 288)@30-frames/s encoding. The MLL compression architecture in [13] achieves a 67% bandwidth reduction by truncating the lowest 3 b of the pixels for the IME part, but for the FME and MC parts, the original reference pixels are used and the bandwidth reduction is only 29.4%. In addition, its hardware cost is greater than other algorithms. The MDA & SFL compression scheme in [11] achieves a good compression and hardware performance. Throughput of as much as 3.13 (6.26) Gpixels/s can be attained and the average DRR is 61.9%. Although the previous algorithms achieve different amounts of bandwidth reduction, they cannot reduce memory requirements.

Our previous lossless scheme [16] achieves a DRR of 68.5%, with the clock speeds of as much as 578 MHz for the compressor and 599 MHz for the decompressor. The throughputs of the compressor and decompressor are 1.54 and 0.78 Gpixels/s, respectively. As the frequency of read access to the reference frames is much higher than that of write access in the HEVC prediction core, the decompressor throughput, rather than that of the compressor, creates the bottleneck in the FMR architecture. The throughput of our previous decompressor cannot meet the real-time encoding requirements of SHV (8K) video, even with the Level D reference data reuse scheme.

The highest DRR achieved by this paper's proposed method is 70.6%, which is 2.1%–20.6% higher than that of its predecessors. By using the parallel processing method, the throughput of up to 2.89 (2.26) Gpixels/s is obtained by the compressor (decompressor), which is almost four (three) times that of our previous decompressor. Both the proposed lossy algorithm and the previous lossless algorithm [16] can contribute to the memory storage optimization, but the compression performance of the lossless algorithm is noticeably decreased for high-rate configurations. The analysis of compression performance and DRAM dynamic power consumption of the two algorithms is shown in Table XVII. The experiments show that the proposed lossy algorithm can achieve a stable DRR and memory space reduction that do not deteriorate for high-rate texture-rich pictures. When QP = 12, the improvements of 7.7% (6.5% and 2.7%) in DRR and 16.0% (13.6% and 11.3%) in memory space reduction are achieved by our three mode lossy algorithms in comparison

TABLE XVI

COMPARISONS OF THE PROPOSED ARCHITECTURE WITH PREVIOUS ARCHITECTURES (QP = {22, 27, 32, 37} AND $\gamma = 0.0002$)

	Lee's [12]	Cheng's [14]	Fan's [13]	Guo's [11]		Lian's [16]		Mode 0		Mode 1		Mode 2	
	com.	com.	com.&dec.	com.	dec.	com.	dec.	com.	dec.	com.	dec.	com.	dec.
Compression type	Lossy	Lossy	Lossy	Lossless		Lossless		Lossy		Lossy		Lossy	
Data reduction ratio(%)	50.0(fixed)	65.3	47.1	61.9		68.5		70.6		69.3		64.9	
CMOS technology	0.18μm	0.18μm	0.13μm	90nm		65nm		65nm		65nm		65nm	
ΔPSNR(dB)	-1.03	-0.10	-0.01	0		0		-0.04		-0.04		-0.05	
Gate count(K)	28.0	26.9	110.4	45.1	34.5	36.5	34.7	24.6	29.5	33.6	38.6	51.4	59.3
On-chip SRAM(byte)	0	512	5120	N/A		192	256	0		0		0	
Max freq.(MHz)	14	10	250	300		578	599	562	578	546	565	543	424
Throughput(pixels/cycle)	2.6	0.45	0.89	10.7	21.3	2.67	1.33	1.33	1.33	2.67	2.67	5.33	5.33
Throughput(Gpixels/s)	0.036	0.0045	0.22	3.13	6.26	1.54	0.78	0.75	0.77	1.46	1.51	2.89	2.26
Memory space save(%)	50	0	0	0		38		41		39		37	
Dynamic power save(%)	50.0	40.3	41.4/18.2*	38.2		56.8		59.3		57.7		54.3	

* 41.4% for the IME part, and 18.2% for the FME and MC parts .

TABLE XVII

COMPRESSION PERFORMANCE AND DRAM DYNAMIC POWER CONSUMPTION ANALYSIS OF OUR PROPOSED LOSSY ALGORITHM AND OUR PREVIOUS LOSSLESS ALGORITHM ($\gamma = 0.0002$)

	QP	Mode 0	Mode 1	Mode 2	Lian's [16]
Data Reduction Ratio (%)	12	63.89	62.70	58.92	56.20
	17	66.90	65.64	61.30	60.43
	22	69.18	67.77	63.24	65.32
	27	70.11	68.76	64.43	67.56
	32	71.03	69.65	65.37	69.50
	37	72.13	70.82	66.56	71.37
Memory Space Reduction (%)	12	37.79	35.41	33.07	21.81
	17	39.30	37.18	34.56	29.25
	22	40.74	38.77	36.28	34.73
	27	41.05	39.02	36.95	36.82
	32	40.78	38.76	37.13	38.48
	37	41.17	39.32	38.19	40.84
Dynamic Power Reduction (%)	12	53.92	52.28	49.05	43.06
	17	56.36	54.77	51.09	48.52
	22	58.32	56.69	52.94	53.63
	27	59.01	57.40	53.93	55.82
	32	59.47	57.85	54.58	57.65
	37	60.30	58.79	55.72	59.71

to our previous lossless algorithm [16]. Because of these improvements, the dynamic power consumption is reduced to 10.9% (9.2% and 6.0%) more by the proposed lossy algorithm than by the previous lossless algorithm.

V. CONCLUSION

This paper presents a lossy FMR algorithm to reduce the external bandwidth and power requirements of the HEVC encoding. The algorithm consists of four parts: a content-aware adaptive quantization, a parallel directional prediction, a dynamic kth-order unary/Exp-Golomb coding, and a partition group table-based storage scheme. As the QP is adopted in the RQL decision, the proposed algorithm achieves a stable high compression performance even in the high-rate configurations. The experiments demonstrate that the savings of 70.6% in bandwidth and 41.0% in memory requirements can be achieved by our algorithm, reducing DRAM dynamic power consumption by 59.3%. The quality change incurred is -0.04 dB in terms of BD-PSNR. Using TSMC 65-nm CMOS technology, our parallel processing architecture can achieve the throughput of up to 2.89 and 2.26 Gpixels/s for compression and decompression, respectively, sufficient to perform SHV(8K)@68-frames/s real-time encoding with the Level D reference data reuse scheme.

REFERENCES

[1] G. J. Sullivan, J.-R. Ohm, W.-J. Han, and T. Wiegand, "Overview of the High Efficiency Video Coding (HEVC) standard," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 22, no. 12, pp. 1649–1668, Dec. 2012.

[2] T. Wiegand, G. J. Sullivan, G. Bjøntegaard, and A. Luthra, "Overview of the H.264/AVC video coding standard," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 13, no. 7, pp. 560–576, Jul. 2003.

[3] T.-Y. Oh *et al.*, "A 7 Gb/s/pin 1 Gbit GDDR5 SDRAM with 2.5 ns bank to bank active time and no bank group restriction," *IEEE J. Solid-State Circuits*, vol. 46, no. 1, pp. 107–118, Jan. 2011.

[4] P.-K. Tsung, W.-Y. Chen, L.-F. Ding, S.-Y. Chien, and L.-G. Chen, "Cache-based integer motion/disparity estimation for quad-HD H.264/AVC and HD multiview video coding," in *Proc. IEEE Int. Symp. Circuits Syst. (ISCAS)*, Apr. 2009, pp. 2013–2016.

[5] C.-Y. Chen, C.-T. Huang, Y.-H. Chen, and L.-G. Chen, "Level C+ data reuse scheme for motion estimation with corresponding coding orders," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 16, no. 4, pp. 553–558, Apr. 2006.

[6] Micron Company, "Calculating memory system power for DDR3," Micron Technol., Inc., Boise, ID, USA, Tech. Rep. TN-41-01, 2007.

[7] Q. Cai, L. Song, G. Li, and N. Ling, "Lossy and lossless intra coding performance evaluation: HEVC, H.264/AVC, JPEG 2000 and JPEG LS," in *Proc. Asia–Pacific Signal Inf. Process. Assoc. Annu. Summit Conf. (APSIPA ASC)*, Dec. 2012, pp. 1–9.

[8] T.-H. Tsai and Y.-H. Lee, "A 6.4 Gbit/s embedded compression codec for memory-efficient applications on advanced-HD specification," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 20, no. 10, pp. 1277–1291, Oct. 2010.

[9] D. Zhou *et al.*, "A 530 Mpixels/s 4096 × 2160@60fps H.264/AVC high profile video decoder chip," *IEEE J. Solid-State Circuits*, vol. 46, no. 4, pp. 777–788, Apr. 2011.

[10] H.-S. Kim, J. Lee, H. Kim, S. Kang, and W. C. Park, "A lossless color image compression architecture using a parallel Golomb–Rice hardware CODEC," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 21, no. 11, pp. 1581–1587, Nov. 2011.

[11] L. Guo, D. Zhou, and S. Goto, "A new reference frame recompression algorithm and its VLSI architecture for UHDTV video codec," *IEEE Trans. Multimedia*, vol. 16, no. 8, pp. 2323–2332, Dec. 2014.

[12] Y. Lee, C.-E. Rhee, and H.-J. Lee, "A new frame recompression algorithm integrated with H.264 video compression," in *Proc. IEEE Int. Symp. Circuits Syst. (ISCAS)*, May 2007, pp. 1621–1624.

[13] Y. Fan, Q. Shang, and X. Zeng, "In-block prediction-based mixed lossy and lossless reference frame recompression for next-generation video encoding," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 25, no. 1, pp. 112–124, Jan. 2015.

[14] C.-C. Cheng, P.-C. Tseng, and L.-G. Chen, "Multimode embedded compression codec engine for power-aware video coding system," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 19, no. 2, pp. 141–150, Feb. 2009.

[15] J. Kim and C.-M. Kyung, "A lossless embedded compression using significant bit truncation for HD video coding," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 20, no. 6, pp. 848–860, Jun. 2010.

[16] X. Lian, Z. Liu, W. Zhou, and Z. Duan, "Lossless frame memory compression using pixel-grain prediction and dynamic order entropy coding," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 26, no. 1, pp. 223–235, Jan. 2016.

[17] E. Y. Lam, "A mathematical analysis of the DCT coefficient distributions for images," *IEEE Trans. Image Process.*, vol. 9, no. 10, pp. 1661–1666, Oct. 2000.

[18] N. Merhav, "Rate–distortion function via minimum mean square error estimation," *IEEE Trans. Inf. Theory*, vol. 57, no. 6, pp. 3196–3206, Jun. 2011.

[19] T. Berger, *Rate Distortion Theory: A Mathematical Basis for Data Compression*. Englewood Cliffs, NJ, USA: Prentice-Hall, 1971.

[20] G. J. Sullivan and T. Wiegand, "Rate-distortion optimization for video compression," *IEEE Signal Process. Mag.*, vol. 15, no. 6, pp. 74–90, Nov. 1998.

[21] T. Wiegand and B. Girod, "Lagrange multiplier selection in hybrid video coder control," in *Proc. IEEE Int. Conf. Image Process. (ICIP)*, vol. 3. Oct. 2001, pp. 542–545.

[22] S. Guo, Z. Liu, G. Li, T. Ikenaga, and D. Wang, "Content-aware write reduction mechanism of 3D stacked phase-change RAM based frame store in H.264 video codec system," *IEICE Trans. Fundam. Electron., Commun. Comput. Sci.*, vol. E96-A, no. 6, pp. 1273–1282, 2013.

[23] F. Sampaio, B. Zatt, M. Shafique, L. Agostini, J. Henkel, and S. Bampi, "Content-adaptive reference frame compression based on intra-frame prediction for multiview video coding," in *Proc. 20th IEEE Int. Conf. Image Process. (ICIP)*, Sep. 2013, pp. 1831–1835.

[24] F. Pan *et al.*, "Fast mode decision algorithm for intraprediction in H.264/AVC video coding," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 15, no. 7, pp. 813–822, Jul. 2005.

[25] S. Xue and B. Oelmann, "Unary prefixed Huffman coding for a group of quantized generalized Gaussian sources," *IEEE Trans. Commun.*, vol. 54, no. 7, pp. 1164–1169, Jul. 2006.

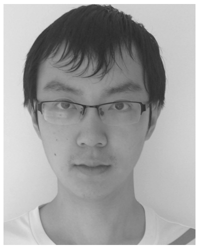
[26] J. Zhou, O. C. Au, and A. Y. Wu, "Secure Exp-Golomb coding using stream cipher," in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process. (ICASSP)*, Apr. 2009, pp. 1457–1460.

[27] *DesignWare Enhanced Universal DDR Memory Controller IP(uMCTL2) Datasheet*, Synopsys, Inc., Mountain View, CA, USA, 2015.

[28] K. McCann, B. Bross, W.-J. Han, I.-K. Kim, K. Sugimoto, and G. J. Sullivan, *High Efficiency Video Coding (HEVC) Test Model 15 (HM15) Encoder Description*, document JCTVC-Q1002, ITU-T/ISO/IEC Joint Collaborative Team on Video Coding (JCT-VC), Apr. 2014.

[29] *CACTI: An Integrated Cache and Memory Access Time, Area, Leakage, and Dynamic Power Model*, accessed on 2008. [Online]. Available: http://www.hpl.hp.com/research/cacti/

[30] O. Vargas, "Achieve minimum power consumption in mobile memory subsystems," *EE Times Asia*, Mar. 2006.



Xiaocong Lian (S'15) received the B.S. degree in electronic science and technology from Northwestern Polytechnical University, Xi'an, China, in 2013, where he is currently pursuing the Ph.D. degree.

His current research interests include algorithms and VLSI implementation for video coding.

Zhenyu Liu (M'07) received the B.E., M.E., and Ph.D. degrees from the Beijing Institute of Technology, China, in 1996, 1999, and 2002, respectively, all in electrical engineering.

From 2002 to 2004, he held a post-doctoral position with Tsinghua University, Beijing, China, where he was involved in the embedded processor architecture design. From 2004 to 2009, he was a Visiting Researcher with the Graduate School of IPS, Waseda University, Shinjuku, Japan. He was with Tsinghua National Laboratory for Information Science and Technology, Research Institute of Information Technology, Tsinghua University, in 2009, where he is currently an Associate Professor. His current research interests include signal processing, energy-efficient real-time video encoding, and application specific processor.

Wei Zhou (M'11) received the B.E., M.S., and Ph.D. degrees from Northwestern Polytechnical University, Xi'an, China, in 2001, 2004, and 2007, respectively.

He is currently an Associate Professor with Northwestern Polytechnical University. His current research interests include video coding and associated VLSI architecture design.

Zhemin Duan received the B.E. and M.S. degrees from Northwestern Polytechnical University, Xi'an, China, in 1978 and 1983, respectively.

He is currently a Professor with Northwestern Polytechnical University. His current research interests include video coding and associated VLSI architecture design.