A CU-level Rate and Distortion Estimation Scheme for RDO of Hardware-friendly HEVC Encoders Using Low-Complexity Integer DCTs

Bumshik Lee, Member, IEEE, and Munchurl Kim, Senior Member, IEEE

Abstract—In this paper, a low complexity CU-level rate and distortion estimation scheme is proposed for HEVC hardware-friendly implementation where a Walsh Hadamard transform (WHT) based low-complexity integer discrete cosine transform (DCT) is employed for distortion estimation. Since HEVC adopts quadtree structures of coding blocks with hierarchical coding depths, it becomes more difficult to estimate accurate rate and distortion values without actually performing transform, quantization, inverse transform, de-quantization and entropy coding. Furthermore, DCT for rate-distortion optimization (RDO) is computationally high because it requires a number of multiplication and addition operations for various transform block sizes of 4-, 8-, 16 and 32-orders and requires recursive computations to decide the optimal depths of Coding Unit (CU) or Transform Unit (TU). Therefore, full RDO-based encoding is highly complex especially for low-power implementations of HEVC encoders. In this paper, a rate and distortion estimation scheme is proposed in CU levels based on a low-complexity integer DCT that can be computed in terms of WHT whose coefficients are produced in prediction stages. For rate and distortion estimation in CU levels, two orthogonal matrices of 4×4 and 8×8 which are applied to WHT are newly designed in a butterfly structure only with addition and shift operations. By applying the integer DCT based on the WHT and newly designed transforms in each CU block, the texture rate can precisely be estimated after quantization using the number of non-zero quantized coefficients and the distortion can also be precisely estimated in transform domain without de-quantization and inverse transform required. In addition, a non-texture rate estimation is proposed by using a pseudo entropy code to obtain accurate total rate estimates. The proposed rate and distortion estimation scheme can effectively be used for HW-friendly implementation of HEVC encoders with 9.8% loss over HEVC full RDO, which much less than 20.3% and 30.2% loss of a conventional approach and Hadamard-only scheme, respectively.

Index Terms—HEVC, Rate-distortion optimization, integer DCT, Hadamard transform, rate estimation, distortion estimation.

This work was supported by the National Research Foundation of Korea (NRF) grant funded by Korea government (MEST) (No. 2014R1A2A2A01006642). Bumshik Lee and Munchurl Kim* are with School of Electrical Engineering in Korea Advanced Institute of Science and Technology (KAIST), 291 Daehak-ro, Yuseong-gu, Daejeon, 34141, Korea.

e-mail: lbs@kaist.ac.kr, mkim@ee.kaist.ac.kr, *: corresponding author
Tel : +82-42-350-7419, Fax : +82-42-350-7619

Fig. 1. An example of CU, PU and TU.

I. INTRODUCTION

Recently, the JCT-VC (Joint Collaborative Team on Video Coding) co-established by ISO/IEC and ITU-T has finalized a new video coding standard, called High Efficiency Video Coding (HEVC) [1]. The HEVC aims at achieving the coding efficiency improvement of about 50% or more compared to H.264/AVC [2] which has been known as one of the video codecs with the best coding performance. Owing to more flexible coding structure with hierarchical quadtree blocks and expanded DCT, the HEVC significantly improves coding efficiency for video sequences from low to high picture resolutions. The HEVC is composed of three main parts to process encoding, which are Coding Unit (CU), Transform Unit (TU) and Prediction Unit (CU). Basic coding processes including motion estimation (ME) and motion compensation (MC), transform, quantization and entropy coding etc. are performed in each CU block. The CU with the maximum size is called the Coding Tree Unit (CTU) for which its size and the number of predefined depths are signaled in a sequence level. The ME/MC is performed in PU, and transform/quantization is performed in TU. The TU is a block unit for transform, quantization, de-quantization and inverse transform at a leaf node of a CU block whose size can range from 4×4 to 32×32, which should not exceed the size of the CU block. The quantized coefficients are scanned in each transform block and input to an entropy coder such as Context-based Adaptive Binary Arithmetic Coding (CABAC) [3]. Fig. 1 shows an example of CU, PU and TU where CU is partitioned in a quadtree manner. In Fig. 1, the l-th CTU (CTU_l) indicates the largest CU, CU_k is the CU in depth level k. The DCT and CABAC entropy coding are performed in each TU, and the optimal CU and TU block sizes are determined in an RDO sense.
where the block (TU or CU) with the least rate-distortion (RD) costs is selected as an optimal block size [4]. In order to obtain the RD costs, the DCT and entropy coding must be recursively performed in each CU depth level. Fig. 2 depicts the computation of RD costs in an HEVC encoder.

![Fig. 2. Rate and distortion optimization on the hybrid video codec.](image)

As shown in Fig. 2, the transform (denoted as T) is DCT-based integer transform which is applied to the predicted residue obtained via ME and MC followed by the quantization (denoted as Q), and the resulting distortion can then be calculated between the original pixel block (S) and the reconstructed pixel block (C) after quantization, de-quantization and inverse transform. The entropy coder (CABAC) for quantized coefficients produces output bitstreams from which rates can be calculated. However, the RDO process in Fig. 2 requires significantly high computational power because the DCT is performed for quadtree structured variable-sized TU blocks of 4x4 to 32x32 in accordance with quadtree structured CU blocks and the CABAC entropy coder causes to significantly increases the complexity of the HEVC encoder. Although such complex DCT and entropy coding based RDO can provide high coding performances, it is not feasible for low-powered HEVC HW encoders with limited resources. Therefore, it is worthwhile to develop an efficient rate and distortion estimation scheme for HW-friendly RDO, not by relying on the full RDO-based encoding.

In this paper, we proposed a HW-friendly rate and distortion estimation scheme in CU levels for which a new low-complexity integer DCT is designed based on WHT. Based on the proposed low-complexity integer DCT, rates and distortions are precisely estimated in CU levels, which is appropriate for RDO of HW-friendly HEVC encoders in the sense that (i) we reuse the results of Hadamard transform obtained from prediction stages like sub-pel ME so that the total complexity of DCT computation is significantly reduced; and (ii) floating points and exponent operations are avoided in rate and distortion estimations.

This paper is organized as follows: In Section II, we briefly describe the previous related works; In Section III, we review the property of DCT expressed in terms of WHT; We design a new low-complexity integer DCT and show how to estimate the distortions and rates in CU levels for RDO in Section IV; In Section V, the experimental results are presented to show the effectiveness of the proposed rate and distortion estimation scheme for HW-friendly HEVC encoders; Finally, the paper is concluded in Section VI.

II. RELATED WORKS

Much research has been conducted to reduce the computational complexities of the video encoders. First, fast mode decision methods for intra- or inter-coding were proposed for HEVC [5-13]. These methods decide particular coding modes in earlier stages or skip them by analyzing the characteristics of the predicted residuals. However, computational powers for non-skipped modes are still necessary to obtain the RD costs by performing DCT and entropy coding. As another approach, many studies on rate and distortion estimation were carried out for H.264/AVC [14-18] and some recent works have been reported for HEVC [19, 20].

Tu et al. proposed rate and distortion models for inter-predictive coding [15] where rates are estimated based on the number of non-zero quantized coefficients derived from the entropy and the distortions are estimated as a mean square error (MSE) based on the proposed distortion model. Both rate and distortion estimates are obtained under the assumption that the transformed residuals are modeled by a Laplacian probability density function (PDF). Then, the Laplacian model parameters in transform domain are derived from the mean absolute difference (MAD) in the pixel domain. However, in spite of elimination of DCT and entropy coding for RDO, it is difficult to apply this scheme to HW implementation because of a number of exponent values and floating-point operations in the models. Furthermore, the Laplacian model parameters in transform domain computed from pixel domain are no longer precise in large CU or TU blocks of HEVC. In [14], [16] and [17], rate estimation schemes for CAVLC (Context-based Adaptive Variable Length Coding) [21] were proposed using the number of quantized coefficient and trailing ones in CAVLC by avoiding the actual entropy coding. However, computational burdens still exist since actual DCT should be performed for these schemes. In [18], a rate and distortion estimation scheme was proposed based on statistical modeling of predicted residues. The rate was estimated by assuming that the transformed residues follow generalized Gaussian distribution (GGD). A self-information by GGD is exploited to obtain the rate while the distortion is calculated in DCT transform domain. The proposed scheme is still complex because many PDF parameters of GGD must be estimated and also actual DCT should be performed.

As investigated in the aforementioned conventional schemes for low complexity RDO, a number of schemes were proposed for H.264/AVC while there are few studies for HEVC because it becomes more difficult to accurately estimate the rates and distortions in CU levels for HEVC due to the enlarged transforms and variable coding block sizes. Especially, the rate and distortion schemes for HW-friendly HEVC encoders with limited resources have rarely been proposed so far. In [19, 20], RDO schemes for HEVC intra coding were proposed. In [19], HT instead of DCT was used only for HEVC intra RDO. In [20], a rate-distortion estimation scheme based on HT is proposed for HEVC Intra/Intra RDO. A new rate-distortion model was proposed based on the sum of absolute transformed difference (SATD). However, since two important parameters to decide the accuracy of the model should be decided by pre-encoding with multiple quantization parameter (QP) values, it is not beneficial for low-complexity encoding.
The HEVC adopts integer DCTs of various block sizes from 4x4 to 32x32 for TU, which plays an important role to improve coding efficiency. Although DCT has been widely used owing to its energy compaction capability toward low frequency components and easy implementation, it is still a crucial burden in computational complexity for RDO in HEVC encoders require a number of multiplication and addition operations. There have been plenty of endeavors to reduce the complexity of DCT operations. Compared to matrix computations, the Chen’s method has significantly reduced the number of multiplication and addition operations [22]. In [23], the DCT kernels approximated to integer values were proposed. Chen’s DCT kernels approximated in integer values are implemented in the HEVC reference software (HM) [37]. The butterfly structure of Chen’s DCT in [22] is replaced with a lifting scheme only with shift and addition operations, which is called BinDCT [24], where it is reported that comparable coding performances are shown with the reduction of the complexity. In [24], a DCT implementation scheme expressed by WHT was proposed. The WHT and the lifting scheme without multiplication operations are used for implementation of the DCT, so called IntDCT [25, 26]. The coefficients in the BinDCT and IntDCT use fractional numbers to express the rotational angles only with shift and addition operations. However, both BinDCT and IntDCT still require a large number of arithmetic operations with approximation errors compared to the original DCTs. Furthermore, it is reported that IntDCT and BinDCT cannot be used for lossless image coding due to a dynamic range problem [28].

In order to overcome the aforementioned limitations, we propose a new low-complexity integer DCT based on WHT and a particular orthogonal transform (denoted as A) which can be implemented in a butterfly structure with shift and addition operations of integer coefficients. Based on the proposed low-complexity integer DCT, the distortions can precisely be estimated in transform domain for CU blocks and the rates for the CU blocks can also be precisely predicted based on its non-zero quantized coefficients after quantization. The contributions of this paper are summarized as follows:

- We newly design an orthogonal transform matrix A to form an integer DCT kernel from a WHT without multiplications.
- We proposed a rate and distortion estimation scheme using matrix A. Consequently, once HT or WHT coefficients are available, the rates and distortions can easily be estimated by the proposed low-complexity DCT that consists of the matrix A and WHT.
- A non-texture rate estimation scheme is also proposed without actual entropy coding, which is used with the proposed texture rate estimate scheme for total rate estimation.

III. DCT EXPRESSED IN TERMS OF WALSH HADAMARD TRANSFORM

Coefficients of HT kernels are composed of integers 1 or -1, and their all row basis vectors are orthogonal [29]. An M-order HT is defined in [29] as

\[
H_M = H_2 \otimes H_{M/2} = \begin{bmatrix} H_{M/2} & H_{M/2} \\ H_{M/2} & -H_{M/2} \end{bmatrix}
\]

(1)

where \( \otimes \) is a Kronecker product, \( H_1 \) and \( H_2 \) are also defined as

\[
H_1 = [1], \quad H_2 = \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix}
\]

(2)

For example, hence, an 8-order HT is

\[
H_8 = \begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ -1 & -1 & -1 & -1 & 1 & 1 & 1 & -1 \\ 1 & -1 & -1 & -1 & 1 & -1 & -1 & 1 \\ 1 & 1 & 1 & -1 & -1 & -1 & 1 & -1 \\ 1 & -1 & -1 & 1 & 1 & -1 & -1 & -1 \\ 1 & 1 & -1 & 1 & -1 & 1 & -1 & 1 \\ 1 & 1 & 1 & 1 & -1 & -1 & -1 & -1 \\ 1 & -1 & -1 & -1 & 1 & 1 & 1 & 1 \end{bmatrix}
\]

(3)

In [30], DCT can be expressed in terms of WHT. In other words, DCT can be written as

\[
C_M = \frac{1}{\sqrt{M}} A_M H_{w,M}
\]

(4)

where \( H_{w,M} \) is an \( M \)-order WHT. \( A_M \) is an \( M \)-order matrix producing a DCT matrix, and can be factorized as [25]

\[
A = B \cdot T \cdot B^T
\]

(5)

where B is a bit reversal matrix which rearranges the input data into the bit-reversed order. For 8-point inputs, B can be expressed as

\[
B = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \end{bmatrix}
\]

(6)

and T is a matrix composed of identity and rotation matrices. An 8-order T is expressed as

\[
T_8 = \begin{bmatrix} I_2 & 0 \\ 0 & U_4 \end{bmatrix}
\]

(7)

where \( I_2 \) is a 2x2 identity matrix , and \( U_2 \) is a \(-\pi/8\) rotation matrix. \( U_4 \) is also a rotation matrix with \( 7\pi/16 \) and \( 3\pi/16 \) rotational angles, and is defined as

\[
U_4 = \begin{bmatrix} \cos \left( \frac{7\pi}{16} \right) & 0 & 0 & -\sin \left( \frac{7\pi}{16} \right) \\ 0 & \cos \left( \frac{3\pi}{16} \right) & -\sin \left( \frac{3\pi}{16} \right) & 0 \\ 0 & \sin \left( \frac{3\pi}{16} \right) & \cos \left( \frac{7\pi}{16} \right) & 0 \\ \sin \left( \frac{7\pi}{16} \right) & 0 & 0 & \cos \left( \frac{7\pi}{16} \right) \end{bmatrix}
\]
For DCT expressed in terms of WHT in (4), it is noted that $H_{w,M}$ is not a Hadamard-ordered matrix but a Walsh-ordered matrix. The orders of the row bases between HT and WHT matrices are different. To obtain the WHT matrix from the Hadamard-ordered matrix, a bit reversal matrix is applied to the Hadamard matrix. In HM reference software [37], all Hadamard-ordered matrices are used for sub-pel ME or intra prediction, etc. However, two matrices produce the identical sum of absolute transformed difference (SATD). Thus, we replace HT ($H_M$) with WHT ($H_{w,M}$) in the HM reference software.

For general expression, $T_N$ of $N$-order is represented as

$$T_N = \begin{bmatrix} T_{N/2} & 0 \\ 0 & U_{N/2} \end{bmatrix}$$

where the initial matrices $T_2$ and $U_2$ are initially defined as a $2\times2$ identity matrix and a rotational matrix with the rotation angle of $-\pi / 8$, respectively. We mainly deal with the $8\times8$ and $4\times4$ DCT expressed in WHT in this paper.

### IV. PROPOSED RATES AND DISTORTION ESTIMATION SCHEME

The relation between DCT and WHT in (4) indicates that DCT can be expressed in terms of WHT. The HM reference software uses Chen’ DCT [22] that requires 16 multiplications and 26 additions in a butterfly structure instead of matrix computations with 52 multiplications and 26 additions for the case of $8\times8$ block size. Although Chen’s DCT is intended for use of low complexity, it is difficult to implement it in resource limited HW environments when the transform kernel size is increased and repetitive computations are performed in the quadtree block structure in RDO of HEVC. We propose a novel rate and distortion estimation scheme based on the property that DCT can be implemented using the WHT. Fig. 2 depicts the overall proposed rate and distortion estimation scheme for HW-friendly HEVC encoders. In Fig. 3, $H_{w}$ is WHT which is performed in the prediction stage for each PU.

The matrix $A$ is applied to the best WHT coefficients to obtain DCT transformed coefficients. Therefore, the coefficients by matrix $H_{w}$ are reused to perform DCT. In addition, quantization and normalization are performed for the matrices $A$ and $H_{w}$. The CU-level rate estimates are obtained after quantization. In addition, distortion is calculated in transform domain without performing inverse transform so low complexity RDO becomes possible.

#### A. Proposed matrix $A$ for low-complexity integer DCT

The matrix $A$ is factorized into bit reversal and rotation matrices as (8). Since the bit reversal matrix can be implemented only with hardwired connection of input and output, computational powers are not necessarily required. However, $U_{4}$ is composed with two rotation matrices and a bit reversal matrix. So, the matrix $A$ needs to be carefully designed so that the increase in complexity is minimized. We propose a matrix $A$ according to the design principles below:

(a) The floating-point rotation coefficients are approximated to integer values for HW-friendly implementation.

(b) The approximated integer values should be expressed only with the powers of 2 or additions of the powers of 2 to avoid multiplication operations.

(c) The matrix coefficients should be maintained as small as possible such that the dynamic ranges are within 16 bits when transform, quantization and normalization are performed.

The design principle in (b) allows avoiding multiplication operations. For instance, multiplication of 12 to input can be implemented only with shift and addition operations. Fig. 4 illustrates the shift and addition operations for the multiplication of an integer 12 to input.

![Butterfly implementation for multiplication of an integer value 12.](image)

Based on the design principles in (a)-(c), we proposed $U_{4}$ matrix for $A_{s}$ and $U_{2}$ matrix for $A_{4}$ as

$$U_{4} = \begin{bmatrix} 2 & 0 & 0 & -8 \\ 0 & 8 & 0 & 0 \\ 0 & 6 & 8 & 0 \\ 8 & 0 & 0 & 2 \end{bmatrix} \begin{bmatrix} 0 & -1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & -1 & 0 \end{bmatrix} \begin{bmatrix} 4 & 0 & -8 & 0 \\ 0 & 4 & 0 & -8 \\ 8 & 0 & 4 & 0 \\ 0 & 8 & 0 & 4 \end{bmatrix}$$

$$U_{2} = \begin{bmatrix} 12 & 4 \\ -4 & 12 \end{bmatrix}$$

(10) and (11) are the matrices with integer coefficients values by considering the design principles in (a)-(c), which can be realized with a butterfly structure. Fig. 5 shows the butterfly structure of $U_{4}$ in (8).
As shown in Fig. 5, the matrix $U_4$ can be implemented without multiplication operations. $U_2$ can also be implemented in a butterfly structure with integer values 12 and 4. The resulting $A_8$ and $A_4$ matrices using (10) and (11) are defined as

$$
A_8 = \begin{bmatrix}
1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 64 & 0 & 32 & 0 & -8 & 0 & 16 \\
0 & 0 & 12 & 0 & 0 & 0 & 4 & 0 \\
0 & -32 & 0 & 64 & 0 & 48 & 0 & 24 \\
0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\
0 & 24 & 0 & -48 & 0 & 64 & 0 & 32 \\
0 & 0 & -4 & 0 & 0 & 0 & 12 & 0 \\
0 & -16 & 0 & -8 & 0 & -32 & 0 & 64
\end{bmatrix}
$$

(12)

Compared to $A_8$, $A_4$ is composed of a single angle ($-\pi/8$) rotation matrix, resulting in a simpler butterfly structure than $A_8$. Fig. 6 depicts the 1-D butterfly structure of $A_8$. As shown in Fig. 6, all coefficients are integer values in the butterfly structure and $A_4$ matrix is embedded in the matrix $A_8$. As a result, the DCT can be implemented using $A_8$ and WHT. If the butterfly structure in Fig. 6 is used for an 8x8 DCT, only 12 additions and 17 shifts are needed. In this case, 24 additions are not included for DCT computations because the WHT coefficients have already been obtained in the prediction stage as aforementioned.
This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TIP.2016.2579559, IEEE Transactions on Image Processing

Table I shows the complexity comparison for various transform structures of 8×8 DCTs. Compared to other DCT structures, our proposed DCT reduces computational operations of multiplications, additions and shifts. Especially, since the DCT by the proposed matrix $A_k$ is performed using the WHT coefficients, 24 additions due to 8×8 Hadamard transform can be saved. Shift operations in HW-implementation can be easily implemented with a concatenation of zeroes to the right or left side of the variable. On the other hand, addition operations require obviously more computational power than that of shift [26]. In our proposed scheme, DCT can be performed only using 16 additions and 24 shifts because HT is already computed in the ME stage. Thus, our proposed scheme requires much less computational powers in number of operation.

Table I. COMPARISON OF COMPLEXITY FOR VARIOUS TRANSFORMS FOR 8×8 DCT COMPUTATION

<table>
<thead>
<tr>
<th>Operation</th>
<th>DCT</th>
<th>Chen [21]</th>
<th>IntDCT [25]</th>
<th>BinDCT [27]</th>
<th>HT</th>
<th>Prop. $\left(A_k\right)$</th>
</tr>
</thead>
<tbody>
<tr>
<td>Mul.</td>
<td>52</td>
<td>16</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Add.</td>
<td>26</td>
<td>26</td>
<td>45</td>
<td>40</td>
<td>24</td>
<td>16</td>
</tr>
<tr>
<td>Shift</td>
<td>0</td>
<td>0</td>
<td>17</td>
<td>23</td>
<td>0</td>
<td>24</td>
</tr>
</tbody>
</table>

In order to investigate the effectiveness of the proposed integer DCT, we compute approximation errors ($\epsilon_{app}$) of the proposed integer DCT under the assumption that an $N$-point input signal ($X$) is assumed to be a zero-mean 1-st order autoregressive (AR) field. So, the covariance matrix ($R_{xx}$) of $X$ with correlation coefficients ($\rho$) is given by

$$R_{xx} = \begin{bmatrix} 1 & \rho & \cdots & \rho^{N-1} \\ \rho & 1 & \cdots & \rho^{N-2} \\ \vdots & \vdots & \ddots & \vdots \\ \rho^{N-1} & \rho^{N-2} & \cdots & 1 \end{bmatrix}$$  \hspace{1cm} (14)

The mean squared error between the original DCT ($C_{DCT}$) and an approximated DCT ($C_{app}$) can be calculated as

$$\epsilon_{app} = \frac{1}{N^2} E[e^T e] = \frac{1}{N^2} E[X^T D^T D X]$$

$$= \frac{1}{N^2} E[Tr(DX^T D^T)] = \frac{1}{N^2} Tr(DR_{xx} D^T)$$  \hspace{1cm} (15)

where $D = C_{DCT} - C_{app}$. Note that the variances of transform coefficients are the diagonal elements of $R_{xx}$. Table II exhibits the approximation errors for various DCT transform kernels and a HT kernel.

Table II. APPROXIMATION ERROR ($\epsilon_{app}$) COMPARISONS

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>0.70</td>
<td>5.98×10^4</td>
<td>1.20×10^3</td>
<td>5.54×10^4</td>
<td>3.59×10^4</td>
</tr>
<tr>
<td>0.75</td>
<td>4.96×10^4</td>
<td>9.87×10^4</td>
<td>4.98×10^4</td>
<td>3.26×10^4</td>
</tr>
<tr>
<td>0.80</td>
<td>3.94×10^4</td>
<td>7.93×10^4</td>
<td>4.31×10^4</td>
<td>2.85×10^4</td>
</tr>
<tr>
<td>0.85</td>
<td>2.94×10^4</td>
<td>5.94×10^4</td>
<td>3.51×10^4</td>
<td>2.35×10^4</td>
</tr>
<tr>
<td>0.90</td>
<td>1.95×10^4</td>
<td>3.94×10^4</td>
<td>2.56×10^4</td>
<td>1.72×10^4</td>
</tr>
<tr>
<td>0.95</td>
<td>9.72×10^3</td>
<td>1.95×10^4</td>
<td>1.10×10^4</td>
<td>9.51×10^3</td>
</tr>
</tbody>
</table>

Fig. 7 shows comparison of various kernels for the original DCT, IntDCT [25], BinDCT [27], Suzuki [28] and the proposed DCT. In Fig. 7 and Table II, our proposed kernel shows less approximation errors especially in a low frequency component such as the 2-nd row while larger approximation errors are shown in the 2-nd and 4-th lows of the IntDCT, BinDCT and Suzuki. It should be noted that approximation errors in low frequency components should be kept as small as possible because low frequency components play important roles to achieve more energy compaction into lower frequency components, thus leading to higher coding efficiency [32].

B. CU-level Rate and Distortion Estimation using Proposed low-complexity DCT based on WHT and $A$

We proposed the matrix $A$ to form a new low-complexity integer DCT in conjunction with WHT for RDO of HEVC.
encoder. As described in Section IV-A, our proposed low-complexity integer DCT kernel requires less arithmetic operations and shows lower approximation errors. For the transform coefficients of the proposed low-complexity integer DCT, we perform the quantization process to obtain the quantized coefficient values from which rate and distortion values are estimated in CU levels. In designing the matrix \( \mathbf{A} \) under the design principles, we sacrifice the orthogonality property to obtain smaller approximation errors. In addition, a different \( l_2 \) norm value is used for each basis row, which must be normalized properly in the quantization process. Hence, the quantization and normalization for WHT and matrix \( \mathbf{A} \) should carefully be designed. We use the design rules for a dead-zone plus uniform threshold quantization (DZ+UTQ) [31] as shown in Fig. 8, which has been adopted in the state-of-the-art video coding standards such as H.264/AVC and HEVC.

![Fig. 8. A dead-zone plus uniform threshold quantizer (DZ+UTQ).](image)

The quantization and de-quantization (reconstruction) for DZ+UTQ can be expressed as

\[
Z_q = \text{round} \left( \frac{W_q + f}{q} \right) \cdot \text{sgn}(W_q) \tag{16}
\]

\[
W_q = q \cdot Z_q \tag{17}
\]

In (16), \( Z_q \) indicates a quantized value of the the \((i,j)\)-th input DCT coefficient, \( W_q \) is the \((i,j)\)-th DCT coefficient, \( f \) is a rounding offset parameter to decide a dead-zone size and the location of a representative level in the quantization intervals, \( \text{round()} \) indicates the function to round a value to its nearest integer and \( \text{sgn()} \) returns the sign of the input signal. In (17), \( W_q \) indicates the \((i,j)\)-th de-quantized DCT coefficient. The transform and quantization (or inverse quantization and inverse transform) can be jointly performed with normalization and shift operations instead of division operations [33]. The quantization and reconstruction are given by

\[
\begin{align*}
\tilde{F}_q &= \text{round} \left( \left\| \mathbf{A}_h \cdot \mathbf{H}_w \cdot \mathbf{A}_T \right\| \cdot \left\| \mathbf{A} \right\| \cdot N_w q_k \right) \\
\tilde{H}_r &= \text{round} \left( \left\| \mathbf{A} \right\|^2 \cdot \left\| \mathbf{A} \right\| \cdot N_r q_k \right)
\end{align*}
\]  

(18)

where \( N_w \) is used for normalization of \( \mathbf{H}_w \) and we use \( N_w=8 \) and \( N_r=4 \) for \( 8 \times 8 \) and \( 4 \times 4 \) matrices in this paper, respectively. In addition, de-quantized matrix is given by

\[
\tilde{F}_d_q = \tilde{F}_q \cdot q_k \tag{19}
\]

When inputs in (18) are the transform coefficients by WHT \( \left( \mathbf{H}_w \right) \), (18) can be represented as

\[
\begin{align*}
\tilde{F}_q &= (C \odot Q_k (i,j) + f \cdot 2^{N_q}) \gg N_q \\
\tilde{H}_r &= (\tilde{F}_q \odot R_k (i,j) + 2^{N_r-1}) \gg N_r
\end{align*}
\]  

(20)

where \( \odot \) is an element-by-element multiplication operation, \( k \) is the \( k \)-th quantization parameter, \( N_q \) and \( N_r \) are the integer numbers for bit-shift in quantization and reconstruction, respectively. The parameter \( f \) is a rounding offset for quantization and controls the dead zone size, ranging from 0 to 0.5. \( Q_k (i,j) \) and \( R_k (i,j) \) in (20) include an \( l_2 \) norm of its row basis kernel and a quantization step size. Thus, the relations between quantization and reconstruction matrices \( Q_k (i,j) \) and \( R_k (i,j) \) for the matrix \( \mathbf{A} \) are given by

\[
\begin{align*}
R_k (i,j) &= \text{round} \left( q_k \cdot 2^{N_r} \left/ \left( N_w \left\| \mathbf{A} \right\|^2 \right) \right. \cdot R_k (i,j) \right) \\
Q_k (i,j) &= \text{round} \left( 2^{N_q+N_r} \left/ \left( N_w \left\| \mathbf{A} \right\|^2 \right) \right. \cdot Q_k (i,j) \right)
\end{align*}
\]  

(21)

In HEVC, \( k \)-th quantization step size is defined as \( q_k = q_0(k\%6)-2^{\left\lfloor \text{floor}(k/6) \right\rfloor} \) where \( q_0(k\%6) = \{0.6250, 0.6875, 0.8125, 0.8750, 1.0, 1.1250\} \) and \( k \) is the \( k \)-th QP value. By plugging \( q_k \) into (21), \( Q_k (i,j) \) and \( R_k (i,j) \) in (21) can be rewritten as

\[
\begin{align*}
R_{k\%6} (i,j) &= \text{round} \left( q_0(i\%6)2^{N_r} \left/ \left( N_w \left\| \mathbf{A} \right\|^2 \right) \right. \cdot R_{k\%6} (i,j) \right) \\
Q_{k\%6} (i,j) &= \text{round} \left( 2^{N_q+N_r} \left/ \left( N_w \left\| \mathbf{A} \right\|^2 \right) \right. \cdot Q_{k\%6} (i,j) \right)
\end{align*}
\]  

(22)

Therefore, the quantization and de-quantization for integer DCT coefficients can be performed by

\[
\begin{align*}
\tilde{F}_q &= (C \odot Q_{k\%6} (i,j) + f) \gg (N_q + \text{floor}(Q/6)) \\
\tilde{H}_r &= (\tilde{F}_q \odot R_{k\%6} (i,j)) \ll (\text{floor}(Q/6) + 2^{N_r-1}) \gg N_r
\end{align*}
\]  

(23)

where \( C \) is an integer DCT block in (4), \( \tilde{F}_q \) and \( \tilde{H}_r \) are quantized and de-quantized blocks, respectively, \( N_q \) and \( N_r \) are the bit shift numbers required for quantization and de-quantization, respectively. In our proposed scheme, \( N_q \) and \( N_r \) are set to 16 and 16 respectively. In our quantization for the proposed integer DCT, six quantization matrices are formed by (22) for each \( 4 \times 4 \) or \( 8 \times 8 \) transform.

Once quantization is performed, a rate estimate in a CU block can readily be made. In our proposed rate estimation scheme, the \( \rho \)-domain rate estimation scheme [34], which is well-known and reported as a relatively accurate rate estimation scheme, is exploit by counting the number of non-zero quantized coefficient in a CU block. The number of non-zero coefficients in a CU block is obtained by

\[
N_{nc} = \sum_{k=1}^{4} \sum_{i,j} \phi_{k}^{i,j} \quad \phi_{k}^{i,j} = \begin{cases} 1, & \tilde{F}_q (i,j) \neq 0 \\ 0, & \text{otherwise} \end{cases}
\]  

(24)

where \( N_{nc} \) is the total number of non-zero quantized coefficients in \( k \) blocks of a CU block. In our proposed rate estimation scheme, the block sizes of the proposed integer DCT are \( 4 \times 4 \) and \( 8 \times 8 \). Therefore, \( N_{nc} \) in a CU block is obtained by counting all non-zero quantized coefficients for a non-split CU block or all sub-blocks of a split CU block. For instance, non-zero coefficients are counted for four \( 8 \times 8 \) DCT blocks in a \( 16 \times 16 \) CU block. So, in this case, the CU-level rate estimate for a texture signal is expressed as
where $k$ is the depth level index of CU, $\alpha$ is a scaling constant.

For distortion estimation in a CU block, we perform distortion estimation based on the proposed integer DCT in transform domain. In Fig. 2, the distortion between $F$ and $\hat{F}$ is computed, where $F$ is a transformed block in (4) and $\hat{F}$ is a de-quantized block in (19). The distortion estimation in transform domain has an advantage of avoiding inverse transform, thus resulting in complexity reduction. In Fig. 1, the sum of squared difference (SSD) in transform domain can be computed as [35]

$$SSD = \left\| F - \hat{F}_{dq} \right\|^2$$

where $F$ is a transform block which should be normalized to obtain the SSD by

$$F = (C \otimes Q_{k\leq 6}(i, j)) \cdot q_k + f) >> (N_q + \text{floor}(QP/6)$$

where $f$ is a rounding offset computed as $f = 1 << (N_q + \text{floor}(QP/6) - 1)$. $F_{dq}$ in (26) is a de-quantized block in (19).

For precise rate estimation in CU levels, the non-texture bits must be correctly estimated, which results from side information including motion vectors, mode information, coded block flag (cbf) etc. The portion of non-texture bits tends to dramatically increase as QP values get higher [36]. In our previous works [36], it is reported that the number of non-texture bits becomes larger than that of texture bits for QP $\geq 32$. Hence, the non-texture bits for RDO should be precisely estimated even for high QP values. We also studied in the previous works [35] that the total non-texture bit number is linearly correlated with the coded bit number for motion vector difference ($mvd$). Once we know the coded bit number for $mvd$, the total number of non-texture bits can readily be estimated. To this end, a pseudo CABAC algorithm for $mvd$ encoding is proposed. Fig. 9 shows a flowchart of estimating a coded bit number for $mvd$.

In Fig. 9, one bit is first used to indicate whether or not the magnitude of the vertical or horizontal $mvd$ component is 0 as denoted by $\Theta$ in Fig. 9. For each of the vertical and horizontal $mvd$ components, total two bits are used where one bit indicate the sign and the other bit signals whether or not the magnitude is greater than 1 as denoted by $\bigodot$ in Fig. 9. The bit number of a virtual exponential Golomb code for the magnitude value of the vertical or horizontal $mvd$ component is added as denoted by $\oplus$ in Fig. 9.

Fig. 10 shows the bit estimation results for the bit estimation algorithm of $mvd$ shown in Fig. 9. As shown in Fig. 10, the $mvd$ bit estimation algorithm precisely estimates the actual $mvd$ bits in CU0 and CU1 for RaceHorses sequence of 832×480 encoded at QP = 23 with random access (RA) configuration. Fig. 11 shows the improvement of rate estimation with the non-texture bit estimation algorithm for $mvd$. Fig. 11-(a) depicts the total bit estimation results without the non-texture bit estimation while Fig. 11-(b) shows the improvement of bit estimation accuracy with the non-texture bit estimation.

From the texture bit estimation in (24) and the non-texture bit estimation in Fig. 9, the CU-level rate estimation can be made as

$$\hat{R}_{CU} = \alpha \cdot N_{mvd} + \beta \cdot \hat{b}_{nonTex}$$

where $\alpha$ and $\beta$ are scaling constants. They are empirically found as $\alpha = 7$ and $\beta = 2$ for P-slice, and $\alpha = 8$ and $\beta = 1$ for B-slice from experiments with a different set of ten training sequences with Toy&Calendar (1920×1080), Walk_Couple (1920×1080), Vintage_Car (1920×1080), Night (1280×720), City (1280×720), BigShips (1280×720), Vidyol (1280×720), Vidyol3 (1280×720), Foreman (352×288), Mobile&Calendar (352×288), which yielded the best result for various QP values and all training sequences. Note that the empirically found $\alpha$ and $\beta$ values are fixed and used for the experiments with the whole HEVC test sequences in Table IV.

Fig. 12 and Fig. 13 show the proposed distortion and rate estimation performances when the CU-level distortion estimation in (26) and rate estimation in (28) are used. Actual distortion and rate values are obtained using the full RDO with the original HM DCT and entropy coders for CU blocks. As shown in Fig. 12 and Fig. 13, the predicted rates and distortions
are highly correlated with their actual values with $R^2 > 0.90$. Unlike the full RDO, since the proposed RDO uses only 8×8 and 4×4 integer DCT blocks based on WHT and A, the estimation errors are a little observed when the distortion and rate values are comparatively large.

However, the resulting coding gain improvement was negligible. In order to use 16×16 DCT based on 16×16 WHT and $A_6$, it is necessary to additionally implement 16×16 WHT for ME which increases computational complexity in the ME stage while it does not increase coding gains compared to those of 4×4 and 8×8 DCT cases in our experiments. In addition, the model parameters $\alpha$ and $\beta$ in (28) are decided as fixed values using training sequences with various characteristics and should optimally be decided under consideration of QP values and CU sizes for the 16×16 case. Thus, only 4×4 and 8×8 DCT blocks are designed and used for RDO in the compromise between coding efficiency and complexity. It should be noted that the proposed low-complexity integer DCTs are only used for RDO. Once the final CU depth is determined, the corresponding HM DCT is performed to produce the resulting bitstreams. Table III shows DCT block types for RDO using the proposed 4×4 and 8×8 integer DCTs for different CU block sizes.

As aforementioned, 4×4 and 8×8 HTs in (1) are used for PU in HEVC. However, the HTs should be aligned in WHT orders for row basis vectors. In order to obtain the DCT in (4), it is necessary to replace HT with WHT matrices for the prediction in PU or to apply a bit reversal matrix to reorder row basis vectors of HT. The bit reversal matrices to obtain 4×4 and 8×8 WHTs are defined as

$$B_4 = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \quad \text{and} \quad B_8 = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \end{bmatrix}$$

Table III. DCT BLOCK TYPEs FOR RDO USING THE PROPOSED 4×4 AND 8×8 INTEGER DCTs FOR DIFFERENT CU BLOCK SIZES

<table>
<thead>
<tr>
<th>CU block sizes</th>
<th>Blocks performed by the proposed DCT</th>
</tr>
</thead>
<tbody>
<tr>
<td>64×64</td>
<td>Sixty four 8×8 blocks</td>
</tr>
<tr>
<td>32×32</td>
<td>Sixteen 8×8 blocks</td>
</tr>
<tr>
<td>16×16</td>
<td>Four 8×8 blocks</td>
</tr>
<tr>
<td>8×8</td>
<td>Four 4×4 blocks</td>
</tr>
</tbody>
</table>

C. Overall RDO scheme based on the proposed rate and distortion estimation scheme and some implementation issues

A full RDO in HM uses various block-sized integer DCTs of 4×4 to 32×32 while 4×4 or 8×8 integer DCTs is used for the proposed RDO in each CU block. We also tried an additional new 16×16 integer DCT based on 16×16 WHT and $A_{16}$.

Table IV. COMPARISONS OF RD PERFORMANCES IN TERMS OF BDBR (%) IN TERM OF BDBR.

<table>
<thead>
<tr>
<th>Sequences</th>
<th>Proposed</th>
<th>Proposed w/o nonTex</th>
<th>Tu [15]</th>
<th>SATD only</th>
<th>Proposed</th>
<th>Proposed w/o nonTex</th>
<th>Tu [15]</th>
<th>SATD only</th>
</tr>
</thead>
<tbody>
<tr>
<td>Class A</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>PeopleOnStreet</td>
<td>9.4</td>
<td>10.4</td>
<td>11.1</td>
<td>26.2</td>
<td>9.4</td>
<td>15.4</td>
<td>16.6</td>
<td>30.8</td>
</tr>
<tr>
<td>Traffic</td>
<td>7.8</td>
<td>10.0</td>
<td>27.7</td>
<td>28.7</td>
<td>8.0</td>
<td>22.5</td>
<td>29.7</td>
<td>29.7</td>
</tr>
<tr>
<td>Class B</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Cactus</td>
<td>16.1</td>
<td>22.1</td>
<td>34.1</td>
<td>39.5</td>
<td>10.0</td>
<td>17.7</td>
<td>19.2</td>
<td>36.8</td>
</tr>
</tbody>
</table>
Table 1: PSNR performance (dB) of various RDO schemes. The average results show a slight improvement compared to the proposed approach.

<table>
<thead>
<tr>
<th>Class</th>
<th>Image</th>
<th>Proposal</th>
<th>HM12.0</th>
<th>SATD Only</th>
<th>Tri's</th>
<th>Proposed</th>
<th>(w/oTex)</th>
</tr>
</thead>
<tbody>
<tr>
<td>E</td>
<td>BQTerrace</td>
<td>13.2</td>
<td>13.6</td>
<td>26.4</td>
<td>25.4</td>
<td>9.0</td>
<td>8.1</td>
</tr>
<tr>
<td></td>
<td>Kimono</td>
<td>16.6</td>
<td>23.5</td>
<td>27.3</td>
<td>42.3</td>
<td>12.8</td>
<td>25.6</td>
</tr>
<tr>
<td></td>
<td>ParkScene</td>
<td>11.0</td>
<td>14.1</td>
<td>24.3</td>
<td>28.3</td>
<td>8.5</td>
<td>11.3</td>
</tr>
<tr>
<td></td>
<td>BasketballDrive</td>
<td>16.3</td>
<td>24.9</td>
<td>33.1</td>
<td>37.1</td>
<td>13.0</td>
<td>23.7</td>
</tr>
<tr>
<td></td>
<td>FourPeople</td>
<td>8.6</td>
<td>11.2</td>
<td>33.4</td>
<td>41.8</td>
<td>5.8</td>
<td>8.0</td>
</tr>
<tr>
<td></td>
<td>Jonny</td>
<td>12.2</td>
<td>14.0</td>
<td>43.9</td>
<td>52.2</td>
<td>5.3</td>
<td>7.2</td>
</tr>
<tr>
<td></td>
<td>KristenAndSara</td>
<td>8.5</td>
<td>10.9</td>
<td>31.5</td>
<td>40.8</td>
<td>5.4</td>
<td>7.7</td>
</tr>
<tr>
<td>C</td>
<td>PartyScene</td>
<td>8.5</td>
<td>10.2</td>
<td>11.3</td>
<td>16.5</td>
<td>7.8</td>
<td>10.2</td>
</tr>
<tr>
<td></td>
<td>BQMall</td>
<td>11.2</td>
<td>13.8</td>
<td>19.2</td>
<td>28.6</td>
<td>9.8</td>
<td>14.1</td>
</tr>
<tr>
<td></td>
<td>BasketballDrive</td>
<td>9.4</td>
<td>12.7</td>
<td>22.5</td>
<td>20.8</td>
<td>12.9</td>
<td>17.1</td>
</tr>
<tr>
<td></td>
<td>RaceHorses</td>
<td>8.3</td>
<td>10.3</td>
<td>11.0</td>
<td>17.7</td>
<td>10.4</td>
<td>14.0</td>
</tr>
<tr>
<td></td>
<td>BlowingBubbles</td>
<td>9.7</td>
<td>13.2</td>
<td>17.7</td>
<td>25.3</td>
<td>7.9</td>
<td>11.4</td>
</tr>
<tr>
<td></td>
<td>BQSquare</td>
<td>8.8</td>
<td>8.9</td>
<td>14.0</td>
<td>20.8</td>
<td>7.6</td>
<td>5.7</td>
</tr>
<tr>
<td></td>
<td>RaceHorses</td>
<td>8.4</td>
<td>10.1</td>
<td>11.1</td>
<td>18.4</td>
<td>8.8</td>
<td>13.8</td>
</tr>
<tr>
<td></td>
<td>BasketballPass</td>
<td>9.1</td>
<td>9.6</td>
<td>15.1</td>
<td>19.8</td>
<td>8.7</td>
<td>13.1</td>
</tr>
<tr>
<td>D</td>
<td>BlowingBubbles</td>
<td>9.7</td>
<td>13.2</td>
<td>17.7</td>
<td>25.3</td>
<td>7.9</td>
<td>11.4</td>
</tr>
<tr>
<td></td>
<td>BQSquare</td>
<td>8.8</td>
<td>8.9</td>
<td>14.0</td>
<td>20.8</td>
<td>7.6</td>
<td>5.7</td>
</tr>
<tr>
<td></td>
<td>RaceHorses</td>
<td>8.4</td>
<td>10.1</td>
<td>11.1</td>
<td>18.4</td>
<td>8.8</td>
<td>13.8</td>
</tr>
<tr>
<td></td>
<td>BasketballPass</td>
<td>9.1</td>
<td>9.6</td>
<td>15.1</td>
<td>19.8</td>
<td>8.7</td>
<td>13.1</td>
</tr>
<tr>
<td></td>
<td>Average</td>
<td>10.7</td>
<td>13.5</td>
<td>23.0</td>
<td>29.5</td>
<td>8.95</td>
<td>12.9</td>
</tr>
</tbody>
</table>

Fig. 14. RD graphs for various RDO schemes under RA configurations.
Fig. 15. Reconstructed images with CU partition information for various RDO schemes when QP=28 is used.

Fig. 16. Subjective quality comparisons (BlowingBubbles (416x240, 10-th frames).

V. EXPERIMENTAL RESULTS

The proposed HW-friendly RDO scheme for HEVC is verified in the HEVC reference software, HM12.0 [37]. For the experiments, eighteen video sequences of various spatial resolutions and signal characteristics from Class A to Class E...
used in JCT-VC are tested. The configurations for GOP (Group of Pictures) structures are set to Low Delay P and Random Access. The number of CU quadtree depths is set to 4 which indicates that CU block sizes range from 64×64 to 8×8, while the TU depth is set to 1 which means that TU block sizes are identical to CU block sizes except the 64×64 CU block where four 32×32 TU blocks are used. This configuration of TU depth 1 is often used for HW-friendly experimental setups. The proposed RDO scheme is compared with the full RDO scheme of HM12.0, Tu’s scheme [15] and SATD-only method in terms of BDBR (%). The values in the Table IV indicates the loss compared to HM12.0 reference software.

Table IV shows the experimental results for the proposed RDO compared to the full RDO scheme of HM12.0, Tu’s scheme and the SATD-only method. Since the proposed RDO does not perform full RDO using all DCT kernels and actual entropy coders, BDBR losses occur in the experiments. Nevertheless, the proposed RDO scheme based on the HW-friendly CU-level rate and distortion estimation outperforms Tu’s scheme and SATD only methods as shown in Table IV. Tu’s scheme was developed for H.264/AVC and transform domain statistics are derived from pixel domain statistics under the assumption that input signals are modeled as a 1-st order AR field. In the SATD-only method, distortions are obtained by SATD and rates are used as fixed values. The SATD-only method is often used for HW-friendly implementation due to low complexity. However, the resulting RD performance is significantly degraded compared to the full RDO as shown in Table IV. The proposed RDO scheme significantly outperforms the two conventional methods for sequences of HD size and above. For Traffic, Kimono and Jonny sequences etc., RD performances are more significantly improved. Table IV also shows the performances of the non-texture bit estimation. As shown in Table IV, RD performance improvements are additionally obtained with less BDBR loss of 2.8% point and 3.95% point in average for Low Delay P and Random Access configurations, respectively, when bit estimation for non-texture is performed. It provides the evidence that the proposed non-texture bit estimation helps to improve the performance of the proposed scheme.

Fig. 14 shows the RD graphs for various RDO schemes. Although Tu’s RDO method improves the RD performance compared to the SATD-only method, the improvement of RD performance is limited as shown in Fig. 14.

Fig. 15 depicts the reconstructed images of cropped sizes with CU partitions for various RDO schemes. Tu’s RDO scheme tends to excessively split CU blocks, thus resulting in many CU blocks of smaller sizes while the SATD-only method is less likely to split CU blocks into lower depth levels. Compared to the two conventional methods, our proposed RDO scheme shows more balanced and very similar CU partitions compared to the HM RDO scheme. Partitions of CU blocks influence BDBR performance and subjective quality of the reconstructed image. Table V also shows the statistics of CU coding types for BasketballDrill (832×480) sequences when QP 28 is used. The numbers in Table V indicate relative areas of CU for the whole frames.

<table>
<thead>
<tr>
<th>RDO Schemes</th>
<th>SKIP</th>
<th>Inter-coded CU</th>
<th>Intra-coded CU</th>
</tr>
</thead>
<tbody>
<tr>
<td>HM12.0</td>
<td>33.4</td>
<td>60.1</td>
<td>6.5</td>
</tr>
<tr>
<td>Proposed</td>
<td>20.3</td>
<td>74.3</td>
<td>5.4</td>
</tr>
<tr>
<td>SATD-Only</td>
<td>4.0</td>
<td>93.8</td>
<td>2.3</td>
</tr>
<tr>
<td>Tu’s</td>
<td>57.9</td>
<td>33.6</td>
<td>8.6</td>
</tr>
</tbody>
</table>

The portions of SKIP mode by the SATD-only method are significantly small, leading to the increase in decoding complexity when the bitstream encoded by SATD-only method is decoded. On the other hand, Tu’s RDO scheme yields large portions of SKIP mode, leading to degradation of RD performances. Compared to the two conventional approaches, the proposed RDO scheme produces similar portions of CU coding modes compared to the HM RDO.

Fig. 16 shows the subjective quality comparisons of various RDO schemes for a cropped region in the 10th reconstructed image of BlowingBubbles (416×240) sequences encoded at QP=33. It is shown in Fig. 16-(b) and -(c) that the reconstructed image region by the proposed RDO scheme is much more visually pleasing than that by the SATD-only method at similar bit amounts around 21Kbits. Furthermore, the proposed RDO scheme produce about 7% less bit amount than Tu’s RDO scheme at almost identical PSNR values as shown in Fig. 16-(b) and Fig. 16-(d).

VI. CONCLUSIONS

Integer DCT plays an important role in RDO-based HEVC coding performance. However, it requires a large computational complexity, which is a burden for HEVC-friendly hardware encoders. Therefore, in this paper, we utilize 4×4 and 8×8 WHT coefficients that can be obtained from prediction stages of HEVC encoding to obtain their corresponding integer DCT coefficients without direct computation of integer DCTs. Inspired from the fact that DCT can be expressed in terms of WHT, we design new 4×4 and 8×8 matrices to form the DCT with WHT. All coefficients values of the newly designed 4×4 and 8×8 matrices are composed of integer values and can be implemented in butterfly structures, yielding the reduction of arithmetic operations by avoiding multiplications. That is, DCT coefficients can be obtained by multiplication-free operations from already available WHT coefficients, which produces a new low-complexity integer DCTs. Based on the new low-complexity integer DCTs, we proposed rate and distortion estimation schemes for HW-friendly RDO of HEVC encoding where the quantization and de-quantization are performed in all integer-arithmetic operations. In addition, a non-texture bit estimation scheme in CU levels is proposed for more elaborate total bit estimation. In order to avoid actual entropy coding for non-texture bit estimations, a pseudo CABAC for motion vector difference is exploit for the algorithm. The experimental results show that the proposed RDO scheme reduces the average BDBR up to 22.35%, compared to the conventional DRO schemes.

REFERENCES

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TIP.2016.2579559, IEEE Transactions on Image Processing

L. Shen, Z. Liu, X. Zhang, W. Zhao, and Z. Zhang,

H. Zhang and Z. Ma,

G. J. Lee, S. Kim, K. Lim, and S. Lee, “A Fast CU Size Decision

D. Marpe, H. Schwarz and T. Wiegand,

Y.-K. Tu, J.-F. Yang and M.-J. Xiong, H. Li, Q.Wu, and F.


