Efficient Bit Rate Transcoding for High Efficiency Video Coding

Luong Pham Van, *Student Member, IEEE*, Johan De Praeter, *Student Member, IEEE*, Glenn Van Wallendael, Sebastiaan Van Leuven, Jan De Cock, *Member, IEEE*, and Rik Van de Walle, *Member, IEEE*

***Abstract*—High efficiency video coding (HEVC) shows a significant advance in compression efficiency and is considered to be the successor of H.264/AVC. To incorporate the HEVC standard into real-life network applications and a diversity of other applications, efficient bit rate adaptation (transrating) algorithms are required. A current problem of transrating for HEVC is the high computational complexity associated with the encoder part of such a cascaded pixel domain transcoder. This paper focuses on deriving an optimal strategy for reducing the transcoding complexity with a complexity-scalable scheme. We propose different transcoding techniques which are able to reduce the transcoding complexity in both CU and PU optimization levels. At the CU level, CUs can be evaluated in top-to-bottom or bottom-to-top flows, in which the coding information of the input video stream is utilized to reduce the number of evaluations or to early terminate certain evaluations. At the PU level, the PU candidates are adaptively selected based on the probability of PU sizes and the co-located input PU partitioning. Moreover, with the use of different proposed methods, a complexity-scalable transrating scheme can be achieved. Furthermore, the transcoding complexity can be effectively controlled by the machine learning based approach. Simulations show that the proposed techniques provide a superior transcoding performance compared to the state-of-the-art related works. Additionally, the proposed methods can achieve a range of trade-offs between transrating complexity and coding performance. From the proposed schemes, the fastest approach is able to reduce the complexity by 82% while keeping the bitrate loss below 3%.**

***Index Terms*—Bit rate adaptation, complexity scalable transcoding, high efficiency video coding (HEVC), machine learning.**

I. Introduction

THE recently finalized High Efficiency Video Coding (HEVC) standard [1], jointly developed by the ITU-T Video Coding Experts Group (VCEG) and the ISO/IEC Moving Picture Experts Group (MPEG), has shown significant advances in compression efficiency [2], [3] compared to the prevalent H.264/AVC coding standard. At the same subjective quality, HEVC saves approximately 50% bit rate. Due to its high compression efficiency, HEVC is expected to be widely used for many applications. These include high quality digital video applications such as the distribution of (ultra) high definition TV signals over satellite, cable, and terrestrial transmission systems; low bit rate applications such as video delivery to mobile devices; and video conferencing. To incorporate the newly developed HEVC standard in such a diversity of use cases where a single video stream cannot match the requirements of all devices and networks, efficient transcoding algorithms are required.

Transrating, which refers to bit rate transcoding within the same video format, is widely used for digital video adaptation and distribution. Transrating allows a video bitstream (i) to adjust to network bandwidth constraints and/or (ii) to meet constraints of the receiver terminal. In the first case, when video is transmitted over networks with fluctuating bandwidth, temporary capacity problems can occur. Having a video bitrate higher than the network bandwidth results in visual artifacts caused by packet loss. To reduce such visual distortions, the video stream has to be scaled to a lower rate in a controlled manner. Using transrating, such bitrate adaptation can follow the constraints of the network in an optimal way. In the second case, a video stream might be stored and/or streamed to various devices with different playback capabilities. With such a diversity of devices, a single copy of encoded video cannot match the requirements of all devices. A possible solution is to store several copies of the video on the server and to send the bitstream that best satisfies the requirements of the user. However, the storage cost of the server would be very high and the pre-encoded video stream may still not exactly match the user requirements. To tackle this problem, the video may be encoded at a high bitrate followed by an online transrating step to meet the requirements of the user device.

Transcoding operations can be categorized into either open-loop transcoding or closed-loop cascaded pixel domain transcoding. For the open-loop transcoder, typically only transformed coefficients in the frequency domain are requantized while the motion parameters are not re-evaluated. Therefore, a mismatch between encoder and decoder reference frames occurs, which results in drift. Several solutions for bit rate scaling by requantization have been investigated for MPEG-1/2 and H.264/AVC bitstreams [4]–[9]. In [5], the variable-length code words corresponding to the quantized DCT coefficients are extracted from the video bitstream. These quantized coefficients are inverse quantized and then simply requantized to match the new bit rate. An alternative to requantization, which is called DCT coefficient dropping or dynamic rate shaping, directly cuts high-frequency coefficients from each macroblock [6]. More recently, different bit rate adaption methods have been proposed for H.264/AVC bitstreams [7]–[10]. An efficient mixed

Manuscript received April 3, 2015; revised September 25, 2015; accepted December 14, 2015. Date of publication December 24, 2015; date of current version February 18, 2016. This work was supported by Ghent University—iMinds, Agency for Innovation by Science and Technology (IWT) Ph.D. and post-doctoral fellow grants, the Fund for Scientific Research-Flanders (FWO-Flanders), and the European Union. The associate editor coordinating the review of this manuscript and approving it for publication was Prof. Yap-Peng Tan.

The authors are with the Department of Electronics and Information Systems, Ghent University—iMinds, Ghent 9000 Belgium (e-mail: luong.phamvan@ugent.be; johan.depraeter@ugent.be; glenn.vanwallendael@ugent.be; sebastiaan.vanleuven@ugent.be; jan.decock@ugent.be; rik.vandewalle@ugent.be).

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/TMM.2015.2512231

transrating architecture containing a low complexity scheme combined with a drift cancelling closed-loop scheme was proposed in [8]. A model-based transrating scheme using requantization of the transform coefficients, which is integrated in a rate control mechanism, has been examined in [9]. The advantage of an open-loop transcoder is its low complexity, but the drift effects result in visual quality losses due to the error propagation. This drift can be reduced in a closed-loop cascaded pixel domain transcoder where the video bitstream is decoded and re-encoded to match the target bit rate. However, a high complexity is required due to extensive re-encoding computations.

In this paper, we propose several techniques to reduce the computational complexity of *closed-loop* transrating for HEVC. A high bit rate input bitstream is decoded and the reconstructed sequence is then re-encoded at a lower bit rate. In this cascade of decoder and encoder, the complexity of transrating is high due to the flexibility of HEVC and the resulting large search space that is evaluated by the encoder. The basic block in HEVC, which is 64×64 pixels in size and is known as the coding tree unit (CTU), is recursively split into coding units (CUs) for which three prediction modes (i.e., skip, inter, and intra modes) are supported [11], [12]. This splitting process is performed for CUs from depth 0 (64×64 pixel CU) to depth 3 (8×8 pixel CU). Each CU is the root for further evaluation of the prediction unit (PU), and transform units. Depending on the mode, for each 2Nx2N block, eight PU sizes can be chosen (four symmetric partitions: 2Nx2N, Nx2N, 2NxN, and NxN and four asymmetric partitions: nLx2N, nRxN, 2NxnD, and 2NxnU). To obtain the most efficient mode for a CU at depth d, all PU partitions and all Residual Quad-Tree (RQT) configurations are evaluated during the rate-distortion optimization (RDO) process. This RDO evaluation decides the most optimal mode by minimizing the RD cost function (J) given by (1) where D is the distortion of the reconstructed CU, λ the Lagrangian multiplier and R the rate required to signal the prediction and the residual information of the CU

$$J = D + \lambda R. \tag{1}$$

This RDO process is recursively performed in the four sub-CUs at the depth $d + 1$ [1]. After evaluating the RD cost of a CU at depth d and the combined RD costs of its sub-CUs, splitting of the CU is decided and signalled by a split-flag.

The proposed techniques reduce the transrating complexity at the CU and PU levels. For the CU evaluation, four fast approaches, which are classified into two classes, i.e., top-to-bottom (T2B) and bottom-to-top (B2T), are proposed. In the first class, where CUs are evaluated from lower depths $(d = 0)$ to higher depths $(d = 3)$, two techniques are proposed. The first technique (T2B) simply re-uses the CU structure from the input bitstream and modifies the structure by evaluating CUs at lower depths. In addition, a machine learning based method $(\mathrm{T2B_{ML}})$, which requires a training phase to implement, makes use of machine learning tools to exploit the correlation between coding information of the input CUs and coding information of the co-located output CUs. In the second class, which includes two methods B2T and $\mathrm{B2T_{TLP}}$, the flow of evaluating CUs moves upwards from higher depths to lower depths. These methods utilize the coding information of CUs from the incoming bitstream and splitting of neighboring CUs. In the PU evaluation, the number of PU candidates is minimized by referring to the PU size of the input video stream.

The rest of this paper is organized as follows. In Section II, we introduce related works and the novel contributions of this paper. The transrating architecture and methodology are described in Section III. Section IV presents the proposed fast transrating techniques. In Section V, the experimental results of the proposed method are presented, showing 82% complexity reduction compared to an unmodified cascaded decoder-encoder approach. Finally, Section VI provides conclusions and future work.

II. Related Works and Contributions of Our Work

A. Fast Transcoding Techniques

The complexity of a closed-loop transcoder is essentially caused by the huge computation in the encoder part. In order to reduce the computational complexity of the transcoder, many approaches have been proposed to optimize the encoding process. Most of these techniques focus on predicting the coding modes in the output video to early terminate encoding. There are two main directions of optimizing the encoder: 1) fast encoding without considering the coding information of the input video, and 2) utilizing the coding information in the input video to accelerate the encoding.

In the first direction, various techniques have been proposed to accelerate the HEVC encoder [13]–[22]. These techniques use the texture characteristics of video and/or utilize the temporal/spatial correlation in the video to predict the coding information of a CU. In [13], the splitting of a CU in intra-coded frames is decided based on the texture homogeneity of the video in the pixel domain and the splitting of its neighboring CUs. A method based on k nearest neighbors has been proposed to determine the CU splittings [15]. In [16], the depth range of a CU is determined to achieve early termination of CU evaluations. On the other hand, early skip mode and merge mode detection methods have been proposed in [17] . Currently, some researchers have started using machine learning to accelerate the HEVC encoder. Support vector machines have been used in [18]. More recently, a decision tree based method has been proposed to reduce the complexity of the HEVC encoder [19] . Although it is reported that the complexity of the HEVC encoder is reduced by the aforementioned techniques, the reduction is still limited.

An alternative direction which can provide a higher complexity reduction for transcoding is to utilize the correlation between coding information of the input and output video. In [23], the statistical properties of the mode distribution are utilized for fast mode refinement of intra prediction. Similarly, in [24] the statistical properties of the mode distribution and motion vector refinement were exploited to reduce the complexity of inter prediction. Given the differences between HEVC and previous video coding standards in their block structure, motion estimation and residual information coding, traditional transrating techniques cannot be directly applied to HEVC. Therefore, several techniques using machine learning for fast transcoding from

MPEG-2 or H.264/AVC to HEVC have been proposed [25], [26]. Notice that the machine learning based techniques in [25], [26] are used in heterogeneous video transcoding, in which the input and output video are encoded using different standards. More recently, a fast machine learning based transcoding technique has been proposed for transcoding in HEVC [27]. This transcoding technique is used for video composition of multiple HEVC bit streams.

In [28], an alternative to fast transcoding for HEVC has been proposed by means of a control stream. In other words, the sender encodes a video at different qualities. The high quality encoded version is sent to the user. When an adaptation process is needed, the sender sends the encoder decisions of one of the other streams without residual information to support the transcoding process. The major downside of this technique is the limitation of flexibility since the number of encoded versions is fixed. Moreover, this technique results in network overhead for transmitting the extra stream.

Finally, a transrating technique for HEVC based on machine learning has been proposed in [29] . In this transcoder, the quantization parameters in the input HEVC video stream is changed to create a lower bit rate video. A gain of 64% in complexity on average is shown, with a loss of 1.76% in coding performance. However, this method is only optimized for predicting the CU structure.

B. Contribution of This Paper

In this paper, we further optimize the performance of the transcoder proposed in [29] with the following contributions.

Firstly, in order to figure out an efficient method for transrating, we propose several techniques which exploit the coding information correlation to speed up transrating. The correlation can be utilized by using machine learning or non-machine learning approaches to predict the CU structure of output video. Besides that, the spatial and temporal correlation is also used to support this prediction. The complexity of the transcoder can be reduced at the CU and/or PU evaluation level, contrary to only the CU level in [29] .

Secondly, we propose a complexity-scalable transrating scheme for particular practical use cases. Different proposed algorithms provide different complexity reductions and bit rate losses. Moreover, the machine learning based algorithm efficiently controls the complexity of the transcoder with two thresholds. The performance of these proposed techniques is evaluated and compared with the state-of-the-art fast HEVC transcoding approaches in terms of complexity reduction and bit rate penalty. Based on this analysis, our proposed methods outperform these algorithms.

Finally, we propose a novel approach for CU evaluation. This approach recursively merges CUs from smaller CUs to larger CUs. In the traditional evaluation flow, the CUs are evaluated from lower depths to higher depths. In contrast, our proposed method evaluates CU in a reverse way. By using this approach, the optimal splitting behavior of CUs at higher depth is known before evaluating the RD cost at the current depth. This prior-known information may be used for an early termination of CU evaluation. Hence, the complexity of transcoder is reduced.

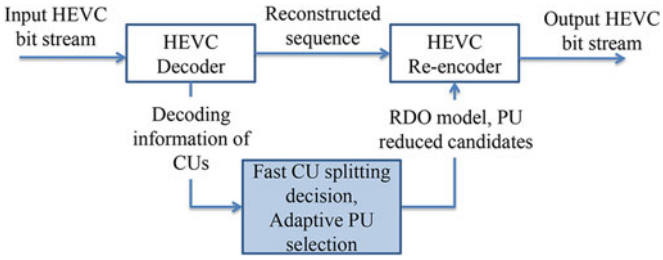

Fig. 1. Proposed transrating architecture.

III. Transrating Architecture and Methodology

The proposed pixel domain transrating architecture is depicted in Fig. 1. The input HEVC bit stream at high quality is first reconstructed. This reconstructed sequence is then re-encoded at the target bit rate. The transrating process focuses on reducing the bit rate of the input bit stream. Therefore, the output bit rate is lower than the input rate. The proposed transrating techniques support both variable bit rate (VBR) and constant bit rate (CBR) schemes.

In the VBR scenario, the input video is encoded using a constant quantization parameter (QP_1). The video is re-encoded using the new quantization parameter QP_2 which is higher than the original QP_1. Therefore, a $\Delta\mathrm{QP}$ is introduced such that $\Delta\mathrm{QP} = \mathrm{QP}_2 - \mathrm{QP}_1$. On the other hand, in the CBR scenario, both input and output video streams are encoded at a constant bit rate. The quantization parameters for each CTU in a frame might be different. They are decided by a rate control algorithm to achieve the target bit rate.

The proposed algorithms optimize the re-encoding process by exploiting the coding information from the input bit stream to limit the amount of CUs and PUs that are evaluated by the encoder. This is indicated in Fig. 1 by the 'Fast CU splitting decision, adaptive PU selection' block. However, it should be noted that in this paper we do not optimize motion estimation.

In order to analyze the correlation (bit rate, CU structure, and PU partition size) between the input and output video streams, conditions for evaluation are specified as follows. The HEVC test model (HM) 7 reference software [30] is used for encoding and decoding. The default common test conditions as defined in [31] are used with the low delay P main configuration (LP) and the VBR scheme. The input and output video streams are both encoded using a coding structure of IPPP . . . with 4 reference frames. The quantization parameter is chosen such that $\mathrm{QP}_1 \in \{22, 27, 32, 37\}$ while $\Delta\mathrm{QP} \in \{2, 4, 6\}$.

The impact of different values of $\Delta\mathrm{QP}$ on the bitrate of the video stream is measured and shown in Table I. As can be expected, there is a larger relative bitrate reduction at higher bit rates (lower QP_1) than at lower bitrates for an equal $\Delta\mathrm{QP}$. In the proposed transrating scheme, we limit the $\Delta\mathrm{QP}$ to 6, resulting in a bit rate reduction by a factor of 2 to 3. As recommended in [32] , further reductions in bit rate should be achieved by other

TABLE I
BIT RATE REDUCTION BY CHANGING QP

Sequence	Bitrate [kbps] (QP_1)	Bit rate reduction[%]		
		$\Delta QP = 2$	$\Delta QP = 4$	$\Delta QP = 6$
ParkScene (1080p)	8365 (22)	-35.16	-54.92	-68.24
	3264 (27)	-32.96	-52.37	-65.65
	1380 (32)	-33.90	-51.63	-64.84
	604 (37)	-34.50	-52.30	-64.62
BasketballPass WQVGA (240p)	1788 (22)	-27.11	-44.55	-57.63
	899 (27)	-27.93	-44.90	-57.19
	453 (32)	-27.29	-42.30	-54.14
	243 (37)	-25.81	-40.20	-50.28
BlowingBubbles WQVGA (240p)	2149 (22)	-30.92	-51.25	-65.25
	880 (27)	-28.71	-48.87	-63.04
	386 (32)	-29.44	-47.91	-61.04
	179 (37)	-28.68	-45.84	-56.59
FourPeople (720p)	2465 (22)	-38.46	-56.51	-67.87
	935 (27)	-27.75	-44.34	-56.54
	469 (32)	-25.15	-39.80	-51.60
	260 (37)	-23.93	-37.93	-48.84
Average		**-29.86**	**-47.23**	**-59.59**

TABLE II
PROBABILITY OF CU DEPTH IN THE OUTPUT VIDEO STREAM GIVEN THE CU DEPTH OF THE INPUT VIDEO STREAM ($P\{d_o|d_i\}$[%])

d_i	$\Delta QP = 2$				$\Delta QP = 6$			
	$d_o = 0$	$d_o = 1$	$d_o = 2$	$d_o = 3$	$d_o = 0$	$d_o = 1$	$d_o = 2$	$d_o = 3$
0	97	3	0	0	98	2	0	0
1	20	76	4	0	43	53	3	0
2	4	20	71	5	16	35	45	4
3	1	7	26	65	6	21	36	36

transcoding methods (e.g., temporal transcoding and/or spatial transcoding).

IV. PROPOSED TRANSCODING SCHEME

In the following subsection the correlation between the coding block sizes (reflected by the depth d of a CU) of the input and output video stream of a cascaded decoder-encoder is evaluated. Afterwards, in Section IV.B, this analysis is used as a starting point for the proposed fast CU splitting techniques. The correlation between the input and output PU sizes is then analyzed in Section IV.C. Section IV.D proposes further accelerations based on adaptive PU evaluation using this PU correlation. Finally, the prediction performance of the proposed methods is evaluated in Section IV.E.

A. Correlation Between Coding Block Depths in Input and Output Streams

In order to evaluate the correlation between the coding block sizes, five video sequences (*ParkScene, BasketballDrill, BQMall, BQSquare, and FourPeople*) are used. The information of the coding depth of CUs encoded at QP_1 and at QP_2 is extracted and shown in Table II, where $P\{d_o \mid d_i\}$ indicates the conditional probability that a CU is re-encoded using depth d_o while it was originally encoded at depth d_i.

In order to obtain the conditional probability, a frame is divided into 8×8 (smallest CU size) blocks. The depth of such an 8×8 block is the depth of the CU covering this block. $P\{d_o \mid d_i\}$ is given by the probability of the output depth of an 8×8 block given the input depth of this 8×8 block.

Table II shows that, when the QP increases for a given d_i, the CU is typically re-encoded at this depth or at a lower depth, corresponding to a larger partition. For instance, when a CU in the input bit stream is encoded using depth 1, there is a probability of 76% that the CU is re-encoded using the same depth and there is a 20% probability that the CU is merged to depth 0, in case of $\Delta QP = 2$. This observation will be exploited in the rest of this paper to reduce the complexity of transrating. When ΔQP is high, the probability that input CUs are merged into CUs at lower depths is high as well. On the other hand, the correlation between coding information of the input and output stream is low when the difference in QP is high.

B. Fast CU Splitting Decision

In this section, four fast transrating techniques are proposed to reduce the complexity of the CU (RDO) evaluation. These methods are classified into two main categories, namely T2B and B2T approaches. In the T2B category, the CU evaluation is performed by a recursive splitting process, where CUs are split from lower depths to higher depths. The trivial T2B applies the CU structure from the input video stream to the output video. An improved T2B approach is a machine learning based method ($\mathrm{T2B_{ML}}$). $\mathrm{T2B_{ML}}$ exploits the correlation of coding information of the co-located CUs to build split-flag decision trees. In the B2T category, CUs are merged from smaller sizes to higher sizes. A first bottom-to-top method is B2T, which applies the CU structure from the input video as the initial structure of the CU in the output video. B2T then merges smaller CUs into a larger CU. To further reduce the complexity of the B2T method, $\mathrm{B2T_{TLP}}$ is proposed. $\mathrm{B2T_{TLP}}$ considers the splitting behavior of the top and left CUs of the current frame, and of the co-located CU in the previously encoded frame.

1) Top to Bottom (T2B) CU Decision: Motivated by the observation in Table II that the CUs in the output video typically have an equal or lower depth compared to the co-located CUs in the input video, the T2B method uses the CTU structure of the input stream to determine the maximum depth it should evaluate for each CU in the output stream.

The T2B technique evaluates the RD cost for CUs from depth 0 to the maximum depth of the initial CTU (iCTu). At each depth of iCTU, the RD cost for every CU of iCTU is evaluated. After checking the RD cost of a CU, the decision to check higher depths is based on the input split flag. If the input CU is split, the split of the output CU is also performed for further evaluation. When the split-flag of the input CU is 0, further splitting of this CU is stopped. It should noted that the RD cost of a CU in iCTU is always evaluated even when the input co-located CU is split. This RD cost evaluation is performed since there is a notable probability that an input CU is re-encoded using a lower depth

Algorithm 1 Pseudo-code for T2B CU evaluation algorithm

Input: initial CTU $iCTU$ = the CTU structure in the input video stream, d_{max} = maximum depth of $iCTU$
for $d = 0$ to d_{max} **do**
  for all $CU_d \in$ CUs at depth d of $iCTU$ **do**
    $RD_{notsplit} = \infty$, $RD_d = \infty$, $SF_d = 0$
    $RD_{notsplit} \leftarrow Encode(CU_d)$, $RD_{split} = \infty$
    if CU_d is split in $iCTU$ **then**
      Go to 4 CUs at depth $d+1$
      $RD_{split} \leftarrow \sum_{i=0}^{i=3} RD_{(d+1)i}$
    end if
    $SF_d \leftarrow (RD_{notsplit} < RD_{split})?0:1$
    $RD_d \leftarrow (SF_d = 0)?RD_{notsplit} : RD_{split}$
  end for
end for
Process the next CTU

TABLE III
DECODED INFORMATION AS INPUT TO DECISION TREES

Parameters	Domain	Meaning
split_flag	0, 1	Split-flag of CU in QP_1
delta_depth	0-3	Difference between the current depth and the max depth of CUs
sum_depth	Number	Sum of depths of CUs
num_pu	Number	Number of PUs
cbf	0, 1	0: If none of the CUs (luminance component) are encoded 1: Otherwise

in the output stream. The RDO process of a CTU in the output video is summarized in Algorithm 1.

2) Machine Learning-Based T2B ($T2B_{ML}$) CU Decision: This method exploits the correlation of coding information of the incoming bit stream and the split-flag of the output CUs. Coding information of co-located CUs from the input bit stream, as listed in Table III, is extracted during transrating. This information is used as the input for machine learning (resulting in decision trees). Out of the decision tree, a split-flag and the corresponding confidence ratio are given. Based on these results, the RDO process for the CU can be controlled.

Decision trees: For each ΔQP, three decision trees indicated as T_0, T_1, and T_2 were constructed. These decision trees predict whether the CUs at depth 0, depth 1, and depth 2, respectively, should be split.

The decision trees have been constructed with machine learning using decoded information (listed in Table III) from four training sequences. These selected four sequences can be classified into three categories: low motion and low complexity (FourPeople), medium activity (ParkScene, and BQSquare) and high motion (BasketballDrill). Originally, the motion vector and residual information in the input video (the variance of the input motion vectors, the means and variances of the residual) have been taken into account for building these decision trees. As these features appear in end-nodes of these trees, the importance of these features is low. Furthermore, these features make the trees more complex. Therefore, these features are ignored to simplify these trees. Table III shows the decoded information, which is used for generating the decision trees. This selected coding information is used as the input of the WEKA data mining tool [33] .

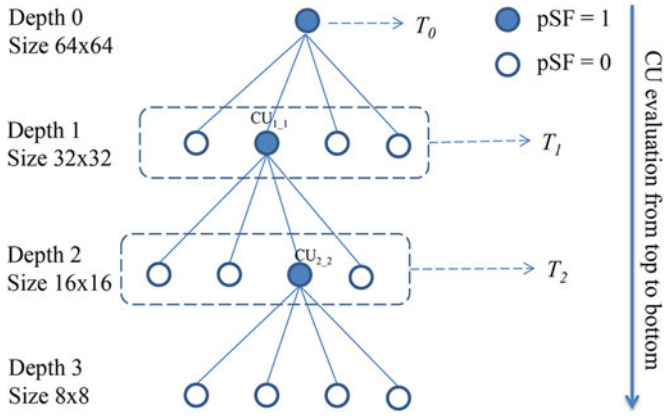

Fig. 2. Splitting ofa CTU using decision trees (pSF = predicted split flag).

The tool used for generating decision trees is *J48*, an implementation of the *C4.5* [34] algorithm in WEKA. This *C4.5* algorithm is a well-known algorithm in the literature for building decision trees and has been widely used in classification applications.

An example of splitting a CU from depth 0 to depth 3 based on decision trees is shown in Fig. 2. The split-flag of the CU at depth 0 (CU_0) is predicted by T_0. The input for this tree is the decoded information from the CUs within a co-located area of 64×64 pixels of this CU_0. The output of this tree is a predicted split-flag (0 or 1), and the probability of the prediction. In this example, CU_0 is decided to be split into four CUs. The splitting of the sub-CUs is decided by tree T_1 using the coded information from CUs at the co-located 32×32 pixels. At depth 1, only CU_{1_1} is decided to be split further while the others are not split. The prediction of the split-flags of the four sub-CUs at depth 2 arising from CU_{1_1} is performed using T_2. This tree predicts a split-flag of 1 for CU_{2_2} and 0 for the other CUs.

RDO evaluation for the transrating process: Based on the decoded syntax information, the decision tree obtained by offline training results in a predicted split-flag. The probability P of the prediction is defined as the ratio of the correctly predicted instances to the total number of instances, and is used here to steer the RDO process. The probability is classified as high, medium, or low by comparing it with two proposed thresholds Thr_1 and Thr_2, in which $Thr_2 > Thr_1$. The summary of classification is given in (2)

$$P \text{ is } \begin{cases} High, & \text{if } P >= Thr_2 \\ Low, & \text{if } P < Thr_1 \\ Medium, & \text{otherwise.} \end{cases} \quad (2)$$

Based on the probability of the prediction, the RDO evaluation is controlled to further refine the predicted split-flag. If P is high, the confidence in the prediction is high. Therefore, the predicted split-flag is directly used as the optimal splitting behavior for the CU. Otherwise, when P is low, the accuracy of

TABLE IV
RD EVALUATION FOR A CU AT DEPTH d [YES (Y) / NO (N)]

Predicted split-flag	P	Check at depth d	Check at depth $d+1$	End recursion
0	High	Y	N	Y
	Medium	Y	Y	Y
	Low	Y	-	N
1	High	N	-	N
	Medium	Y	-	N
	Low	Y	Y	Y

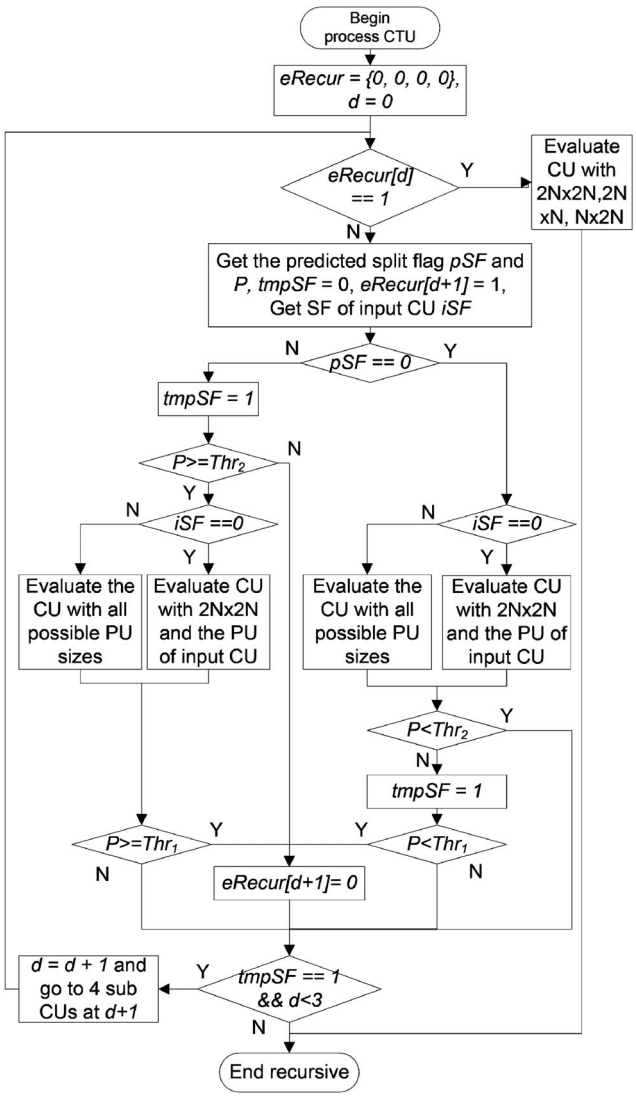

Fig. 3. Flowchart of $\mathrm{T2B_{ML\_PU}}$ algorithm. The "end Recursion"(eRecur) is signalled from depth d to depth $d+1$. Therecursive process is terminated until depth $\mathrm{d} = 3$ or $eRecur = 1$.

the prediction is low and the predicted split-flag is re-evaluated. By adjusting Thr_1 and Thr_2, the transcoding complexity can be controlled and a tradeoff between complexity and coding performance can be achieved. The details of the proposed RDO process are depicted in Table IV and the overall algorithm is summarized in Fig. 3. In this table, 'end recursion' indicates whether the recursive splitting process is terminated after the RD evaluation at depth d and/or $d+1$. When 'end recursion' is 'N', the split flag is predicted again and the RD evaluation process is recursively implemented at depth $d+1$. Otherwise, the RD evaluation process is terminated.

For instance, if the predicted split-flag is 0 and P is high, the CU should not be split and the RD cost evaluation is only performed at the current depth d. When P is medium, the RD cost is evaluated at both depth d and depth $d+1$. However, the CU at depth $d+1$ should not be split, and the recursion is ended. When P is low, the RD cost calculation is performed at depth d and depth $d+1$ without splitting further.

For a predicted split-flag of 1, if P is high, the CU is immediately split to depth $d+1$. Else, the RD cost is evaluated at depth d when P is either medium or low. If P is low, the CU is evaluated at depth $d+1$ and the split recursion is terminated. If P is medium, at depth $d+1$, the split-flag prediction is performed and the RD evaluation process is controlled based on the new predicted split-flag.

3) Bottom-to-Top (B2T) CU Decision: In the top to bottom approach, the evaluation is performed for CUs from lower depths to higher depths. In other words, the RD cost of the CU is normally obtained at lower depths first. Then the RD cost of CUs at higher depths is derived. The splitting of a CU at a lower depth is only decided when both its own RD cost and those of all CUs at higher depths are obtained. As a result, the RD costs at lower depths are always calculated even if the CU will be split. The B2T method is proposed to address this problem. In this approach, the CUs are evaluated from higher depths to lower depths. The optimal splitting behavior of CUs at higher depths is utilized to decide whether the RD cost of CUs at lower depth is obtained and the evaluation is terminated. The details of the proposed method for transcoding are presented as follows.

The co-located CU structure in the input video serves as an initial starting point for the CU structure in the output video. After all, as seen in Section IV.A, the depth of CUs after transrating is usually higher or equal to the depth of the CUs before transrating. The RDO process is recursively performed by merging sub-CUs from the initial depth d_{max} to depth 0. The CU size at the maximum depth (d_{max}) is always evaluated. The merging process is performed conditionally: if all 4 sub-CUs at depth $d+1$ are not split, it might be more optimal to use a larger CU size. Therefore, the RD cost for this CU at depth d is obtained. Otherwise, the CU is not re-evaluated and considered as split. The algorithm of the B2T approach is summarized in Algorithm 2, in which SF_{ndi} and RD_{ndi} are respectively the optimal split flag and RD cost of the i^{th} CU of the four child CUs at depth $d+1$.

An example of splitting a CTU using B2T is given in Fig. 4. At depth 3, RD costs for only 8 CUs, indicated as shaded circles, are obtained, whereas an unmodified encoder might evaluate the costs of 64 CUs. When evaluating at depth 2, B2T evaluates RD costs for 8 CUs. After these RD costs are calculated, RD costs of $\mathrm{CU_{2\_1}}$ and $\mathrm{CU_{2\_7}}$ are compared to the sum of the RD costs of their previously evaluated sub-CUs at depth 3. Assume that $\mathrm{CU_{2\_1}}$ is decided not to be split while the best decision for $\mathrm{CU_{2\_7}}$ is to split. At depth 1, the RD cost of $\mathrm{CU_{1\_4}}$ is not evaluated since the decision has already been made to split

Algorithm 2 Pseudo-code for B2T CU evaluation algorithm

Input: initial CTU $iCTU$ = the CTU structure in the input video stream, d_{max} = maximum depth of $iCTU$
for $d = d_{max}$ to 0 **do**
  for all $CU_d \in$ CUs at depth d of $iCTU$ **do**
    $RD_{notsplit} = \infty$, $RD_d = \infty$, $SF_d = 0$
    $RD_{split} = \infty$
    if CU_d is not split in $iCTU$ **then**
      $RD_{notsplit} \leftarrow Encode(CU_d)$
    else
      $nd \leftarrow d + 1$
      $Ck_d \leftarrow (SF_{nd\_0} \| SF_{nd\_1} \| SF_{nd\_2} \| SF_{nd\_3})$
      if $Ck_d = 0$ **then**
        $RD_{notsplit} \leftarrow Encode(CU_d)$
        $RD_{split} \leftarrow \sum_{i=0}^{i=3} RD_{nd\_i}$
      end if
    end if
    $SF_d \leftarrow (RD_{notsplit} < RD_{split})?0 : 1$
    $RD_d \leftarrow (SF_d = 0)?RD_{notsplit} : RD_{split}$
  end for
end for
Process the next CTU

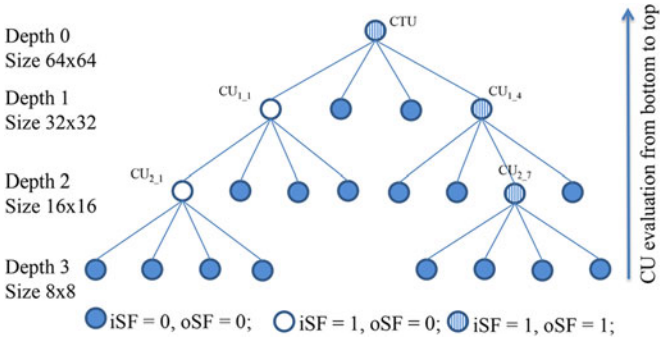

Fig. 4. Example of splitting a CTU based on B2T. iSF and oSF are the split flags of the input andoutput CUs, respectively.

CU_{2_7}. Only the RD cost of CU_{1_1} is evaluated since all of 4 corresponding sub-CUs are not split. The two shaded CUs at this depth are also evaluated. Finally, the RD calculation for the root CU at depth 0 is also skipped since not all sub-CUs can be merged.

4) B2T CU Decision Based on Top, Left CUs, and Co-Located CU in Previous Frame ($B2T_{TLP}$): This method works similar to B2T. However, it considers not only the CU structure from the incoming bit stream but also the splitting behavior of neighboring CUs (namely top and left CUs) and the co-located CU in the previously encoded frame. The RD cost of a CU at a lower depth is obtained if all CUs at higher depths can be merged and the top, left, and co-located CU are not split. Otherwise, the RD cost for this CU is not evaluated.

C. Correlation Between Prediction Unit Sizes in Input and Output Streams

The complexity of selecting the optimal CU size is reduced by the proposed fast CU splitting techniques, which exploit the correlation between CU structures of co-located CUs. However, the correlation between *PU sizes* in the input and output video bit stream was not utilized yet. Here, we first analyze this correlation. Then we propose an adaptive PU selection method which utilizes this correlation to reduce the complexity of the optimal PU partition selection (which is a subprocess of the CU selection process). This proposed method reduces the number of evaluated PU partitions by referring to the PU partition in the input bit stream. Table V shows the correlation between the PU size of co-located CUs in the input and output bit stream. PU_i and PU_o are PU sizes of a CU in the input CU and the co-located CU in the output stream, respectively. The five sequences indicated in Section IV.A are analyzed. $P\{\mathrm{PU}_o \,|\, \mathrm{PU}_i\}$ is the conditional probability that a CU is encoded using PU_o given PU_i when both CUs in the input and output bit streams have the same depth.

It can be clearly seen in Table V that, for any given PU_i, there is a significant probability that PU_o has either the same partitioning or a 2Nx2N partitioning.

Algorithm 3 Pseudo-code for predictive PU evaluation of a CU algorithm using T2B, B2T, and $\mathbf{B2T_{TLP}}$

Input: Split flag iSF of the co-located CU in the input video
if $iSF = 0$ **then**
  Get PU iPU of the co-located CU in the input video
  Evaluate the output CU with iPU and 2Nx2N
else
  Evaluate the output CU with all possible PU sizes
end if
Process the next CU

D. Predictive PU Selection

The proposed adaptive PU selection only evaluates PU partitions having a high probability given the input PU size. For three methods (T2B, B2T, and $\mathrm{B2T_{TLP}}$), if the CU in the input bit stream is not split, only PU sizes 2Nx2N and PU_i are evaluated. Consequently, the number of PU candidates is reduced from 8 to 2. Otherwise if the input CU is split, all possible PU sizes are evaluated for the output CU. The selection algorithm for these hree approaches is summarized in Algorithm 3.

The PU size candidates for a CU in the $\mathrm{T2B_{ML}}$ method depend on the predicted split-flag and the split-flag of the CU in the input. These candidates of CUs at depth d and sub-CUs at depth $d+1$ are jointly controlled by following the RDO model proposed in Table IV. At depth d, if the predicted split flag (pSF) is 0 or $pSF = 1$ with a low or medium accuracy, the PU selection is controlled by the splitting of the input CU. If the input CU is not split, the output CU is evaluated using the input PU (iPU) and 2Nx2N. Else, the CU is evaluated with all possible PU sizes. At depth $d+1$, if the recursion is decided as stop, only the three largest PU sizes including 2Nx2N, 2NxN, and Nx2N are evaluated. Else, the split flag is newly predicted and the PU selection process is recursively performed for depths $d+1$ and $d+2$. The overall flow chart of $\mathrm{T2B_{ML\_PU}}$ is given in Fig. 3.

TABLE V
PROBABILITY OF PU PARTITIONING MODE OF THE OUTPUT VIDEO STREAM GIVEN THE PU PARTITIONING OF THE INPUT VIDEO STREAM

PU_i	$P\{PU_i\}$[%]	$P\{PU_o \mid PU_i\}$[%]							
		2Nx2N	2NxN	Nx2N	NxN	2NxnU	2NxnD	nLx2N	nRx2N
2Nx2N	78.67	**92.47**	2.01	2.65	0.21	0.54	0.61	0.71	0.79
2NxN	5.57	**59.40**	**30.54**	4.38	0.20	2.34	1.26	0.99	0.90
Nx2N	7.06	**54.97**	2.99	**36.26**	0.19	0.70	0.67	2.68	1.53
NxN	1.26	**45.92**	2.91	4.00	**44.32**	0.86	0.40	1.09	0.50
2NxnU	1.78	**61.45**	7.48	3.62	0.10	**23.93**	1.14	1.14	1.14
2NxnD	1.63	**61.94**	5.48	3.55	0.06	1.23	**25.62**	1.10	1.03
nLx2N	2.27	**57.83**	2.38	9.14	0.07	0.73	0.77	**27.78**	1.29
nRx2N	2.07	**61.27**	2.37	7.39	0.05	0.74	0.75	1.58	**25.84**

TABLE VI
SPLIT-FLAG PREDICTION ACCURACY AND THE PU SIZE MATCHING RATE

CFG	Seq	Resolution	Split flag prediction accuracy [%]				PU size matching rate [%]			
			Trivial	T2B	$T2B_{ML}$	$B2T_{TLP}$	Trivial	T2B	$T2B_{ML}$	$B2T_{TLP}$
LP	BQSquare*	416x240	67.55	84.75	82.83	83.21	78.67	87.20	87.62	87.43
	FourPeople*	1280x720	61.27	83.90	82.62	81.80	86.87	91.54	91.31	91.76
	BQMall	832x480	63.46	81.56	80.46	79.55	80.92	88.12	87.99	88.43
	PartyScene	832x480	62.83	81.13	79.48	79.64	72.34	83.37	83.65	83.62
	Kimono	1920x1080	57.42	79.73	79.12	76.87	83.34	90.61	90.21	90.93
	BasketballDrive	1920x1080	60.96	81.66	80.14	79.29	83.86	89.82	89.68	90.09
	Average		**64.08**	**83.60**	**81.63**	**81.10**	**82.51**	**89.32**	**89.30**	**89.58**
RA	BQSquare*	416x240	64.08	90.76	88.86	87.48	85.20	94.30	94.35	93.71
	FourPeople*	1280x720	65.85	90.89	89.77	87.01	89.09	95.29	95.00	94.55
	BQMall	832x480	65.09	87.47	86.33	83.42	83.51	91.94	91.71	91.45
	PartyScene	832x480	60.94	86.99	85.02	83.07	80.24	91.21	91.23	90.19
	Kimono	1920x1080	60.04	86.20	85.39	81.96	87.09	93.81	93.50	93.64
	BasketballDrive	1920x1080	63.37	85.81	84.31	81.80	86.58	92.65	92.44	92.50
	Average		**65.34**	**88.66**	**86.93**	**84.57**	**85.38**	**93.04**	**92.88**	**92.54**

* The sequences in the training set for modelling of $T2B_{ML}$.

E. Prediction Accuracy of the Proposed Techniques

The prediction performance of the proposed methods has been evaluated for the use of VBR as indicated in Section III with a ΔQP of 6. In the CU evaluation, the prediction accuracy is measured for three CU depth levels. At each CU depth, the accuracy of prediction is given by the probability that the split-flags predicted by the proposed methods and an anchor transcoder are the same. In the anchor transcoder, the CU size is decided by the regular HM reference software. At the PU level, the PU size matching rate is the probability that the PU sizes of the proposed methods and the anchor transcoder are the same given that the CU sizes of the proposed and anchor method are the same.

The results are shown in Table VI where the trivial method copies the input coding structure to the output video stream. As can be seen in Table VI, the proposed methods have a high prediction accuracy. There is a remarkable improvement in accuracy of about 20% of our proposed methods compared to the trivial transcoder. Among the proposed methods, T2B provides the best prediction performance with 83.60% and 88.66% for LP and RA configurations, respectively. The prediction accuracy of T2B is high since the CU is always evaluated whenever the input CU is split or not. At the PU evaluation level, the proposed adaptively PU size selection achieves a matching rate of 90% which is about 7% higher than the trivial method.

The result in Table VI shows that the off-line trained model used in $\mathrm{T2B_{ML}}$ is generalized enough. The prediction accuracies for training sequences and test sequences are almost similar. The advantage of using the off-line training model is that a re-training phase is not needed during transcoding. Therefore, the off-line training approach is used in $\mathrm{T2B_{ML}}$.

V. EXPERIMENTAL RESULTS

The proposed methods are evaluated by comparing the performance with several transcoders. These include an unmodified decoder-encoder cascade transrater, the trivial methods and various fast encoding and transcoding algorithms. These trivial methods copy the CU size and/or PU size from the input bit stream to the output bit stream. Firstly, the evaluation conditions are described. Thereafter, the experimental results of Trivial, $\mathrm{T2B_{ML}}$, and $\mathrm{B2T_{TLP}}$ at both the CU and PU levels are analyzed in terms of bit rate increasing and transcoding time. These methods are evaluated in both the variable bit rate (VBR) and constant bit rate (CBR) scenarios. Finally, coding performance of our proposed methods are compared with two state-of-the-art fast HEVC encoding and transcoding algorithms.

A. Evaluating Conditions

In the experiments, all sequences of classes B, C, D, and E listed in [31] excluding the training sequences have been tested. The experiments are tested based on a platform using 64-bit Scientific Linux 6 operating system running on a PC with an integrated Intel dual-socket quad-core 2.27 GHz and 12 GB RAM. The proposed algorithms are implemented in the HEVC test model (HM) 7 reference software [30] under the test conditions as defined in [31]. Search mode "TZSearch" and "FEN" (fast encoder decision) are enabled. In other words, the proposed algorithms are compared with the best speed performance of HM. The CU structure is set with a max CU size of 64×64 pixels and a maximum depth of 4. The performance of the proposed scheme is evaluated in terms of Bjøntegaard Delta Bit rate (BDBR) [35] for both low delay P main (LP) and random access (RA) configurations. For LP, only the first frame is intra-coded while the intra period is set to 32 for RA. In the BDBR measurement, Peak Signal to Noise Ratio (PSNR) calculations between the re-encoded and the original sequence are used. Additionally, complexity reduction, which is measured by the time saving (TS), is given by

$$TS(\%) = \frac{T_{\mathrm{Original}}(ms) - T_{\mathrm{Proposed}}(ms)}{T_{\mathrm{Original}}(ms)}. \quad (3)$$

Here, T_{Proposed} is the total transrating time using the proposed method while T_{Original} is the total transrating time using an unmodified cascaded decoder-encoder setup. Since the same code base is used for the original encoder, the Trivial methods, and all proposed techniques, the difference in time saving gives an indication of the complexity reductions.

B. Coding Performance Under the VBR Scenario

In the VBR scheme, the input video stream is encoded using a constant $\mathrm{QP}_1 \in \{22, 27, 32, 37\}$. This video is reconstructed and coded at a lower bit rate using a higher constant QP_2. The difference of the input and output quantization parameter $\Delta\mathrm{QP}$ is set as $\{2, 4, 6\}$. Firstly, an analyses on the flexible transcoding complexity of $\mathrm{T2B_{ML}}$ using thresholds is provided. Then, the experimental results of all described algorithms are elaborated on.

1) Complexity-Scalable Transcoder Using the Machine Learning Based Method: A trade-off between transcoding complexity and bit rate loss can be achieved by the $\mathrm{T2B_{ML}}$ method with the use of two thresholds. It is clear that, when the proposed thresholds increase, the number of RD evaluations increases accordingly. Consequently, the rate-distortion complexity trade-off of the transrating architecture can be varied with these thresholds. Different pairs of (Thr_1, Thr_2) have been evaluated to come to a usable trade-off. Experimental results using different relevant values are presented in Table VII. When both of these thresholds are high (0.85, 0.90),, the bit rate penalty is very small (0.77%). However, the transrating complexity reduction is then small (53.00%) as well. In contrast, when these two thresholds are small (0.50, 0.75), the bit rate penalty is high with an increase of 2.78% and the complexity reduction is also high (67.36%). When Thr_1 is 0.75 and Thr_2 is 0.85, this method achieves 64.14% complexity reduction with a slight increase of BDBR (1.49%). In the following evaluation, these thresholds (0.75, 0.85) are used as a default for $\mathrm{T2B_{ML}}$.

TABLE VII
CODING PERFORMANCE OF $\mathrm{T2B_{ML}}$ WITH DIFFERENT THRESHOLDS

CFG	Thr_1[%]	Thr_2[%]	BDBR[%]	TS[%]
LP	85	90	0.77	53.00
	75	90	1.14	54.50
	75	85	1.49	64.14
	50	75	2.78	67.36
RA	85	90	0.40	57.20
	75	90	0.65	59.40
	75	85	0.88	63.99
	50	75	1.69	69.00

2) Coding Performance Analyses: The experiment under the LP configuration is analysed first. Then, the performance with the use of the RA configuration is elaborated on.

For the LP configuration, detailed experimental results for each class of the Trivial, $\mathrm{T2B_{ML}}$ and $\mathrm{B2T_{TLP}}$ architectures that optimize the evaluation at both CU and PU level are presented in Table VIII. The comparison of the average performance of these methods is presented in Table IX and visualized in Fig. 5.

As can be seen in Table VIII, the Trivial methods can achieve a low complexity for both CU (Trivial) and PU ($\mathrm{Trivial_{PU}}$) evaluations. On average, Trivial and $\mathrm{Trivial_{PU}}$ can reduce the transrating complexity by 75.65% and 91.12%, respectively. The low complexity of these methods is achieved by directly copying the CU and PU structures from the input bit stream to the output bit stream. However, the simple re-use of the input CU and PU structures results in BDBR losses which strongly increase with rising $\Delta\mathrm{QP}$ values. This could be expected from the probabilities in Tables II and V, which indicate that the CU and PU sizes in the output streams typically become larger for increasing $\Delta\mathrm{QP}$ values. The Trivial methods, however, only evaluate the CUs at the depths of the input CUs and skip the evaluation at lower depths. As a result, the Trivial method increases the bit rate on average by 7.49% while an increase of 15.23% is measured for the $\mathrm{Trivial_{PU}}$ method.

The complexity reductions of $\mathrm{T2B_{ML}}$ and $\mathrm{B2T_{TLP}}$ are smaller than those of the Trivial method. However, these proposed methods significantly outperform the Trivial method in terms of coding performance. The proposed method $\mathrm{T2B_{ML}}$ reduces the complexity of transrating by 64.14% with a 1.49% penalty in bit rate. When the PU evaluation is optimized, the complexity reduction increases to 76.22% with a negligible bit rate increase of about 2.23%. The complexity reduction of $\mathrm{B2T_{TLP}}$ is higher than $\mathrm{T2B_{ML}}$ (66.18% and 79.65% for CU and PU evaluations). $\mathrm{B2T_{TLP}}$ has a loss of coding performance with 1.93% for CU and 2.65% bit rate increase for PU evaluations.

Fig. 6 shows examples of the CU size results obtained by applying our proposed algorithms and the Trivial method. We defined the difference between coding depths (dCU) of a frame obtained by a method and this frame obtained by $\mathrm{HEVC_{Anchor}}$ transrating as the average of absolute depth differences between

TABLE VIII
PERFORMANCE OF DIFFERENT DESCRIBED TECHNIQUES TO A DECODER-ENCODER CASCADE WITH THE VBR SCHEME UNDER THE LP CONFIGURATION

Method	Class	BDBR increase[%]			Time Saving (TS)[%]			Average	
		$\Delta QP = 2$	$\Delta QP = 4$	$\Delta QP = 6$	$\Delta QP = 2$	$\Delta QP = 4$	$\Delta QP = 6$	BDBR	TS
Trivial	B	3.74	7.01	10.94	74.51	75.10	77.01	7.23	75.54
	C	2.88	6.68	11.16	71.98	73.34	73.93	6.90	73.08
	D	2.89	6.11	9.71	72.18	72.00	72.20	6.24	72.13
	E	5.64	9.64	13.89	76.41	78.90	78.69	9.72	78.00
	Avg.	**3.78**	**7.32**	**11.37**	**74.52**	**75.48**	**76.94**	**7.49**	**75.65**
$T2B_{ML}$	B	2.11	1.73	1.38	67.54	63.30	59.38	1.74	63.41
	C	1.36	1.57	1.19	63.95	59.34	53.16	1.27	58.62
	D	1.12	1.29	1.04	61.19	55.72	48.98	1.22	55.30
	E	3.28	1.46	0.71	72.94	72.84	71.43	1.82	72.40
	Avg.	**1.85**	**1.52**	**1.11**	**67.60**	**63.96**	**60.03**	**1.49**	**64.14**
$B2T_{TLP}$	B	1.45	2.29	2.97	65.89	65.59	64.96	2.24	65.48
	C	1.04	2.08	2.93	61.91	62.30	60.71	2.02	61.64
	D	1.09	1.86	2.45	61.43	59.53	58.11	1.80	59.69
	E	1.11	1.57	1.87	71.73	73.90	73.36	1.52	73.00
	Avg.	**1.20**	**1.99**	**2.60**	**66.27**	**66.47**	**65.81**	**1.93**	**66.18**
$Trivial_{PU}$	B	6.39	12.75	19.98	90.78	91.08	91.25	13.04	91.04
	C	6.14	14.78	24.94	90.00	90.36	90.79	15.29	90.39
	D	5.86	13.30	22.20	89.85	89.56	89.99	13.79	89.80
	E	10.72	19.77	29.27	91.56	92.38	92.33	19.92	92.09
	Avg.	**7.18**	**14.88**	**23.64**	**90.80**	**91.19**	**91.36**	**15.23**	**91.12**
$T2B_{ML\_PU}$	B	1.49	2.30	2.88	76.74	75.28	74.39	2.23	75.47
	C	1.78	2.60	2.23	77.73	72.67	65.76	2.20	72.02
	D	1.45	2.25	2.04	74.97	69.77	62.92	1.91	69.10
	E	3.76	2.46	2.02	87.13	84.91	83.51	2.75	85.19
	Avg.	**1.97**	**2.40**	**2.32**	**78.13**	**76.17**	**74.36**	**2.23**	**76.22**
$B2T_{TLP\_PU}$	B	1.68	2.84	3.60	79.53	79.50	78.42	2.71	79.15
	C	1.63	3.16	3.93	76.20	76.22	70.76	2.91	74.40
	D	1.58	2.78	3.52	75.54	73.79	72.19	2.63	73.84
	E	1.71	2.43	2.81	85.29	86.24	85.68	2.31	85.74
	Avg.	**1.65**	**2.80**	**3.48**	**79.96**	**80.15**	**78.84**	**2.65**	**79.65**

TABLE IX
PERFORMANCE OF ALL DESCRIBED TECHNIQUES COMPARED TO A DECODER-ENCODER CASCADE UNDER THE VBR CONDITION

CFG	Method	BDBR increase[%]			Time Saving (TS)[%]			Average	
		$\Delta QP = 2$	$\Delta QP = 4$	$\Delta QP = 4$	$\Delta QP = 2$	$\Delta QP = 4$	$\Delta QP = 6$	BDBR	TS
LP	Trivial	3.78	7.32	11.37	74.52	75.48	76.94	7.49	75.65
	T2B	0.45	0.65	0.62	47.99	50.54	51.34	0.57	49.96
	$T2B_{ML}$	1.85	1.52	1.11	68.00	64.20	60.23	1.49	64.14
	B2T	0.93	1.41	1.81	63.24	61.50	61.79	1.38	62.18
	$Trivial_{PU}$	7.18	14.88	23.64	90.82	91.19	91.36	15.23	91.12
	$T2B_{PU}$	0.99	1.45	1.51	62.82	64.04	64.96	1.32	63.94
	$T2B_{ML\_PU}$	1.97	2.40	2.32	78.13	76.17	74.36	2.23	76.22
	$B2T_{PU}$	1.42	2.27	2.73	77.23	76.31	75.37	2.14	76.30
	$B2T_{TLP\_PU}$	1.65	2.80	3.48	79.96	80.15	78.84	2.65	79.65
RA	Trivial	2.13	4.57	7.29	74.18	74.42	74.63	4.66	74.41
	T2B	0.27	0.32	0.29	50.99	51.66	52.38	0.29	51.67
	$T2B_{ML}$	1.05	0.95	0.64	68.46	63.47	60.04	0.88	63.99
	B2T	0.57	0.95	1.21	64.68	63.51	62.79	0.91	63.66
	$Trivial_{PU}$	3.70	8.62	14.23	90.15	90.24	90.31	8.85	90.24
	$T2B_{PU}$	0.44	0.68	0.66	65.75	66.14	66.76	0.59	66.22
	$T2B_{ML\_PU}$	2.51	2.21	1.48	80.39	80.71	79.22	2.07	80.11
	$B2T_{PU}$	0.72	1.43	1.63	79.15	77.89	76.85	1.26	77.96
	$B2T_{TLP\_PU}$	1.04	2.03	2.77	83.07	82.40	81.71	1.95	82.39

pixels. As can be observed from Fig. 6, the CU structures of $\mathrm{B2T_{TLP}}$ and $\mathrm{T2B_{ML}}$ are very similar to the CU structure obtained by $\mathrm{HEVC_{Anchor}}$ transrating. The $\mathrm{Trivial_{PU}}$ method encodes CUs using higher depths compared to $\mathrm{HEVC_{Anchor}}$ transrating.

When the ΔQP increases, we see different effects on the complexity reduction and coding performance of the $\mathrm{T2B_{ML}}$ and B2T methods.

The correlation between coding information of CUs in the input and output bit stream is reduced when ΔQP increases.

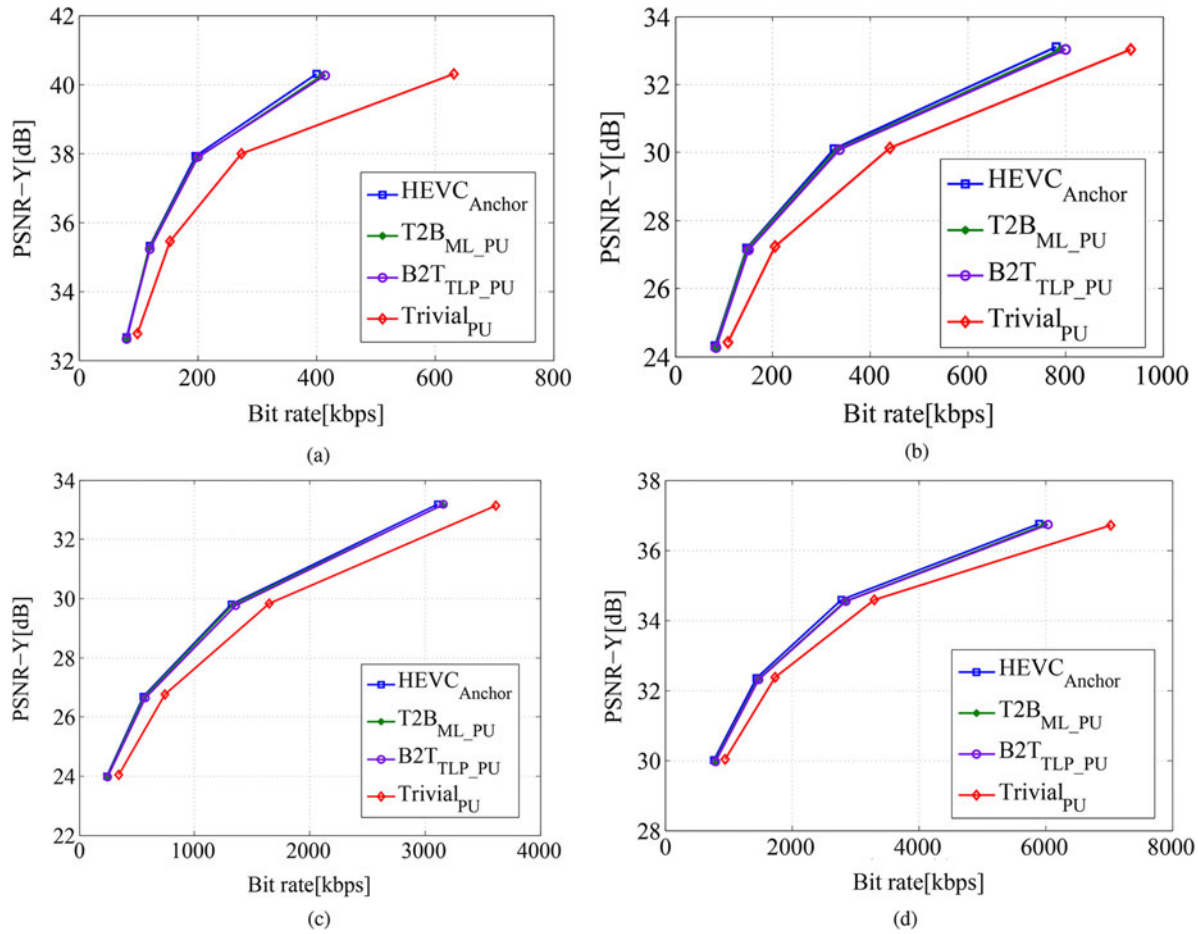

Fig. 5. RD performancefor transrating with a $\Delta \mathrm{QP} = 6$ using the CBR scheme and the LP configuration. The RD performances of our proposed methods very close to the performance of the $\mathrm{HEVC}_{\mathrm{Anchor}}$ transcoder and significantly outperform the performance of the $\mathrm{Trivial}_{\mathrm{PU}}$ transcoder. (a) The Johnny sequence. (b) The BQSquare sequence. (c) The PartyScene sequence. (d) The Basketball Drive sequence.

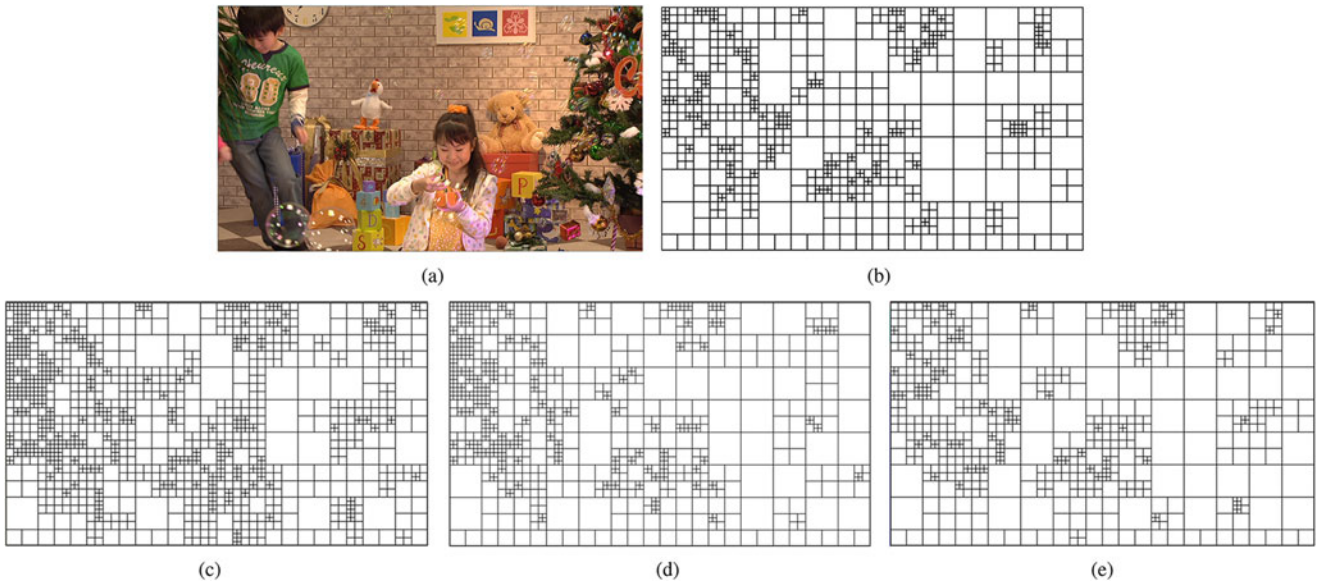

Fig. 6. CU structures generated by the different algorithms for the 200th frame in the PartyScene sequence, $\mathrm{QP}_1 = 32, \Delta \mathrm{QP} = 6$. The VBR and LP configuration are used in this experiment. The CU structures obtained by our proposed methods are similar to one obtained by the $\mathrm{HEVC}_{\mathrm{Anchor}}$ transcoder. (a) Original. (b) Anchor. (c) $\mathrm{Trivial}_{\mathrm{PU}}$ $\mathrm{dCU} = 0.52$. (d) $\mathrm{T2B}_{\mathrm{ML\_PU}}$ $\mathrm{dCU} = 0.33$. (e) $\mathrm{B2T}_{\mathrm{TLP\_PU}}$ $\mathrm{dCU} = 0.27$.

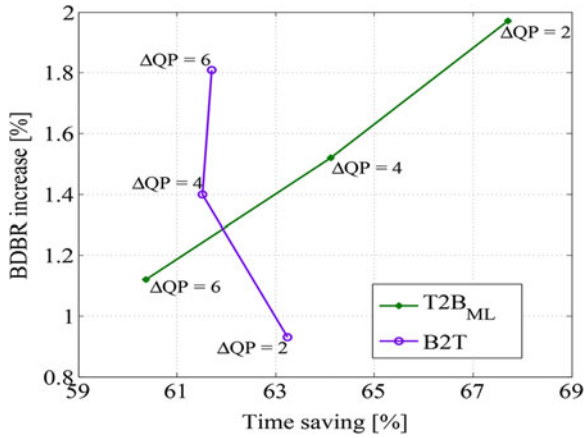

Fig. 7. BDBR increase and time saving of $\mathrm{T2B_{ML}}$ and B2T when ΔQP increase with the use of VBR and the LP configuration.

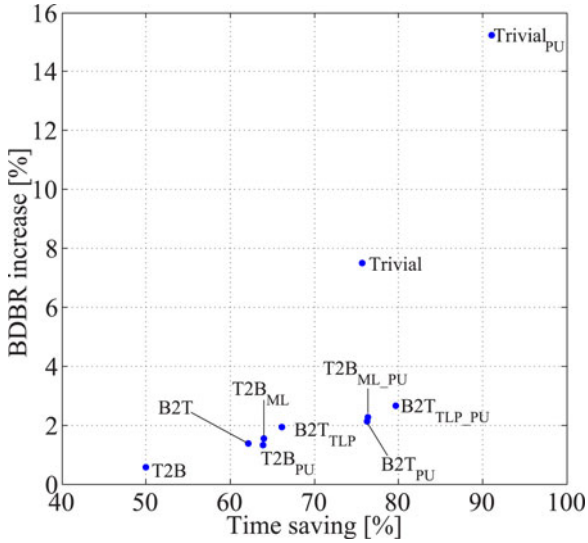

Fig. 8. BDBR increaseand time saving of all described techniques compared to cascaded decoder-encoderwith the use of VBR under the LP configuration.

Consequently, the probability of correct predictions in $\mathrm{T2B_{ML}}$ is reduced. Therefore, the number of CU re-evaluations increases, resulting in a higher transrating complexity (and a reduction of the bit rate penalty). However, the initial CU structure for evaluating a CU in B2T is unchanged when ΔQP increases. As a result, the transrating complexity reduction of B2T is only slightly reduced (and remains around 63%).

Since the difference between the initial CU structure and the optimized CU structure is larger when ΔQP increases, the bit rate penalty of B2T increases with ΔQP. The coding performance comparison of these two methods when ΔQP increases is depicted in Fig. 7. The rate-distortion plots for the methods optimizing the PU evaluation are depicted in Fig. 5. As can be seen, the RD performance of the proposed methods is similar to an unmodified HEVC cascaded decoder-encoder $(\mathrm{HEVC_{Anchor}})$ and clearly better than the $\mathrm{Trivial_{PU}}$ method.

Table IX shows the coding performance of all proposed methods and the Trivial methods. Notice that the difference in performance for each class is not remarkable as demonstrated in Table VIII. Therefore, the average performance of all classes is given in the remainder analysis. The results are summarized visually in Fig. 8.

As can be seen in Fig. 8, a trade-off between coding performance and transrating complexity can be achieved by the proposed methods. Depending on the required complexity reduction, one of the above techniques can be used, such that the highest RD is guaranteed.

For the experiment under the RA configuration, the results is presented in Table XI. These results demonstrate that the proposed algorithm can significantly reduce the transcoding complexity with a negligible bit rate penalty. With a complexity reduction of 51.67%, the T2B method results in a bit rate increase of only 0.29%. For a higher complexity reduction of about 64%, B2T and $\mathrm{T2B_{ML}}$ show the same coding performance with a bit rate increase of 0.9%. At the PU level, T2B can reduce 66% complexity with a bit rate error of only 0.59%, which is smaller than the error of B2T and $\mathrm{T2B_{ML}}$ in the CU evaluation level. Therefore, with the complexity reduction target of 66% (2/3 reduction), $\mathrm{T2B}_{PU}$ is the most advisable solution. However,

TABLE X
INPUT BIT RATE SETUP FOR CBR

Class	Resolution	Bit rate of the input video R_i [kbps]			
		R_{i0}	R_{i1}	R_{i2}	R_{i3}
B	1920x1080	12000	5000	2500	1500
C	832x480	2300	1000	500	300
D	416x240	600	250	100	75
E	1280x720	5000	2000	1000	600

when the transcoding complexity needs to be reduced further, $\mathrm{B2T_{TLP\_PU}}$ and $\mathrm{T2B_{ML\_PU}}$ are more reliable. These methods are able to reduce a notable amount of 80% to 82.4% transcoding complexity with about a 2% bit rate increase. Between these two algorithms, $\mathrm{B2T_{TLP\_PU}}$ is preferred over $\mathrm{T2B_{ML\_PU}}$ regarding the implementation performance since a model and a complex prediction need to be integrated in $\mathrm{T2B_{ML\_PU}}$.

It should be noticed that the proposed algorithms are applied for only inter frames. Doing so, the coding performance loss is recovered at intra frames. As a result, the performance of using RA is better than using LP. For instance, $\mathrm{B2T_{TLP\_PU}}$ under RA results in a higher complexity reduction (82.4%) compared to using LP. However, the bit rate penalty under RA is lower (1.95% compared to 2.65%).

C. Coding Performance Under the CBR Scenario

In practical scenarios, when streamed over the internet, the video may be switched between networks with different bandwidth limitations. In such scenarios, the video is transcoded with the use of a constant bit rate encoder. In this section, the proposed algorithms are evaluated in the following scheme. The input video is encoded at a constant bit rate of R_i (kbps). Afterwards, the video is reconstructed and transcoded to a lower bit rate R_o (kbps) given by (4) where α is the bit rate reduction factor and $0 < \alpha < 0.5$. For higher bit rate reductions, other

TABLE XI
PERFORMANCE OF ALL DESCRIBED TECHNIQUES COMPARED TO A DECODER-ENCODER CASCADE UNDER THE CBR CONDITION

CFG	Method	BDBR increase[%]			Time Saving (TS)[%]			Average	
		$\alpha = 0.85$	$\alpha = 0.70$	$\alpha = 0.55$	$\alpha = 0.85$	$\alpha = 0.70$	$\alpha = 0.55$	BDBR	TS
LP	Trivial	3.87	5.49	9.75	74.31	74.59	74.49	6.37	74.46
	T2B	1.40	1.27	1.89	49.89	50.32	50.64	1.52	50.28
	T2B_ML	2.40	2.12	3.04	68.12	61.84	56.58	2.52	62.18
	B2T	1.81	1.95	0.92	63.80	62.94	61.47	1.56	62.74
	Trivial_PU	5.67	10.58	16.55	90.77	90.97	90.96	10.93	90.90
	T2B_PU	1.70	2.06	2.94	62.64	62.80	62.76	2.24	62.73
	T2BML_PU	2.57	3.82	3.92	81.97	75.34	69.40	3.44	75.57
	B2T_PU	2.22	2.68	2.18	78.12	77.20	75.74	2.36	77.02
	B2TTLP_PU	2.16	3.14	2.33	81.07	80.39	79.23	2.54	80.23
RA	Trivial	5.75	7.01	7.80	73.57	73.59	73.69	6.85	73.61
	T2B	3.00	0.20	0.94	55.45	55.80	56.57	1.38	55.94
	T2B_ML	1.70	2.15	1.82	68.61	62.58	59.84	1.89	63.68
	B2T	2.03	2.64	0.58	64.01	63.02	62.03	1.75	63.02
	Trivial_PU	8.79	12.04	15.48	90.17	90.19	90.23	12.10	90.20
	T2B_PU	1.78	2.70	1.46	68.72	68.70	68.49	1.98	68.64
	T2BML_PU	2.73	3.61	2.45	82.68	76.99	73.34	2.93	77.67
	B2T_PU	1.95	3.30	2.21	78.05	77.87	76.36	2.48	77.43
	B2TTLP_PU	2.80	3.01	2.47	81.81	81.47	80.90	2.76	81.40

transcoding approaches (spatial or temporal transcoding) are suggested

$$R_o = (1 - \alpha) * R_i. \tag{4}$$

Both LP and RA configurations are tested. For BDBR evaluation, the experiment is carried out for four input bit rate values $(\mathrm{R}_{i0}, \mathrm{R}_{i1}, \mathrm{R}_{i2}, \mathrm{R}_{i3})$ as defined in Table X.

Table XI presents the performance of the proposed algorithms under the CBR test condition. In comparison with the use of VBR, the proposed algorithms yield the same complexity reduction. In contrast, the performance is slightly worse in terms of bit rate loss. This loss is generated due to a large range of QP differences between the input and output video. This large QP difference may appear since QPs of the input and output video are independently derived to achieve the input and output rates. Note that the difference in bit rate loss between CBR and VBR is below 1%.

Again, the proposed methods demonstrate a superior performance over the trivial approaches. Among these proposed algorithms, the $\mathrm{B2T}_{\mathrm{TLP\_PU}}$ shows the best performance with a remarkable complexity reduction of about 80% with bit rate losses of 2.54% and 2.76% for LP and RA, respectively.

D. Performance Comparison With the State-of-the-Art

Since transrating for HEVC is a novel topic, to evaluate the performance of the proposed methods, these methods are compared with various fast encoding algorithms [15], [16], [18]–[22], and an HEVC composition transcoder [27] in terms of transrating complexity and bit rate increases. The bit rates are set using a VBR scheme. These fast encoding algorithms are used to encode the reconstructed video of the input stream. It should be noted that, the input coding information is not utilized in these fast encoding references except De Praeter [27]. Therefore, the comparison is not entirely fair. However, the significant performance improvement of our proposed methods implies that the proposed methods are notably efficient in reducing the complexity of an HEVC transcoder. Since each reference work yields different values of BDBR and time saving, we obtain the ratio between BDBR and time saving (B/T) [19]. This parameter shows the amount of BDBR loss per time saving. The lower of B/T means a better performance. The coding performance comparison is shown in Table XII.

TABLE XII
PERFORMANCE COMPARISON WITH RELATED WORKS UNDER VBR IN TERMS OF TIME SAVING (TS) AND BDBR/TS (B/T)

CFG	Optimization	Method	BDBR[%]	TS[%]	B/T
LP	CU	Shen[18]	1.66	42	3.95
		Xiong[15]	1.90	42	4.52
		Lee[20]	1.22	61	2.00
		Ahn[21]	1.00	43	2.33
		De Praeter[27]	2.01	65	3.09
		T2B	**0.57**	**50**	**1.14**
		$\mathrm{T2B_{ML}}$	**1.49**	**64**	**2.33**
		B2T	**1.38**	**62**	**2.23**
	CU + PU	Liquan[16]	1.15	41	2.80
		Liquan[22]	0.88	52	1.69
		T2B	**1.32**	**64**	**2.06**
		$\mathrm{T2B_{ML}}$	**2.23**	**76**	**2.93**
		B2T	**2.14**	**76**	**2.82**
RA	CU	Correa[19]	0.28	37	0.77
		Shen[18]	1.40	45	3.11
		Xiong[15]	2.21	40	5.53
		Lee[20]	1.43	62	2.31
		Ahn[21]	1.40	49	2.86
		T2B	**0.29**	**52**	**0.59**
		$\mathrm{T2B_{ML}}$	**0.88**	**64**	**1.38**
		B2T	**0.91**	**64**	**1.42**
	CU + PU	Correa[19]	1.33	63	2.11
		Liquan[16]	1.50	42	3.57
		Liquan[22]	0.68	49	1.39
		T2B	**0.59**	**66**	**0.89**
		$\mathrm{T2B_{ML}}$	**2.07**	**80**	**2.59**
		B2T	**1.26**	**78**	**1.62**

In general, time saving of our proposed methods is notably higher than the other methods in both the CU and PU

optimization levels. In terms of B/T evaluation, the bit rate loss of our proposed methods is usually lower than or equal to other methods for the same complexity reduction. There are a few exceptions, such as for joint CU and PU optimization under the LP configuration where Liquan [22] yields a lowest B/T. However, this method achieves a complexity reduction of 52% which is much lower than the complexity of $\mathrm{T2B_{ML}}$ and B2T (76%).

VI. Conclusion

In this paper, we proposed several optimized transrating techniques for HEVC. The correlation of coding information of co-located CUs in the input and output video streams was exploited to reduce the complexity of CU and PU evaluations. At the CU level, two options for recursing through the split tree are considered, namely top (lower depths) to bottom (higher depths) or bottom to top. The top to bottom approach (T2B) accelerates the RD cost calculation of a CU by immediate splitting to smaller sizes or by early termination of the recursion, depending on the input CU structure. A more advanced method $(\mathrm{T2B_{ML}})$ predicts the splitting behavior of a CU by using decision trees generated with machine learning techniques. On the other hand, in the bottom to top approach (B2T), the CU structure of input CUs is re-used and recursively evaluated by merging sub-CUs into larger CUs. Additionally, the splitting behavior of neighboring CUs is also considered to reduce the number of RD evaluations in the $\mathrm{B2T_{TLP}}$ method.

Furthermore, during the PU evaluation process, the number of PU candidates is reduced by exploiting information from the input video stream. Experimental results show that the proposed transrating methods maintain the coding efficiency of an unmodified cascaded decoder-encoder, while significantly reducing the transcoder complexity. On the PU evaluation level, $\mathrm{B2T_{TLP}}$ can reduce the complexity of transrating by 82% with a bit rate increase of 1.95%. In addition, by considering all the proposed techniques, trade-offs between transrating complexity and coding performance can be made.

Acknowledgment

This work was carried out using the Stevin Supercomputer Infrastructure at Ghent University.

References

[1] *High Efficiency Video Coding* ITU-T Rec. H.265 and ISO/IEC 23008-2 (HEVC), Apr. 2013.
[2] G.Han, J. Ohm, W.-J. Han, and T. Wiegand, "Overview of the high efficiency video coding (HEVC) standard," *IEEETrans. Circuits Syst. Video Technol.*, vol. 22, no. 12, pp. 1649–1668, Dec. 2012.
[3] J. Ohm, G. Sullivan, H. Schwarz, T. K. Tan, and T. Wiegand, "Comparison of the coding efficiency of video coding standards including high efficiency video coding (HEVC)," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 22, no. 12, pp. 1669–1684, Dec. 2012.
[4] K. Seo, S. Lee, J. Kim, and J. Koh, "Rate control algorithm for fast bit-rate conversion transcoding," *IEEE Trans. Consum. Electron.*, vol. 46, no. 4, pp. 1128–1136, Nov. 2000.
[5] P. Assuncao and M. Ghanbari, "A frequency-domain video transcoder for dynamic bit-rate reduction of MPEG-2 bit streams," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 8, no. 8, pp. 953–967, Dec. 1998.
[6] A. Eleftheriadis and P. Batra, "Dynamic rate shaping of compressed digital video," *IEEE Trans. Multimedia*, vol. 8, no. 2, pp. 297–314, Apr. 2006.
[7] D. Lefol, D. Bull, and N. Canagarajah, "Performance evaluation of transcoding algorithms for H.264," *IEEE Trans. Consum. Electron.*, vol. 52, no. 1, pp. 215–222, Feb. 2006.
[8] J. De Cock, S. Notebaert, P. Lambert, and R. Van de Walle, "Requantization transcoding for H.264/AVC video coding," *Signal Process.: Image Commun.*, vol. 25, pp. 235–254, Apr. 2010.
[9] N. Hait and D. Malah, "Model-based transrating of H.264 coded video," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 19, no. 8, pp. 1129–1142, Aug. 2009.
[10] C. Deknudt, P. Corlay, A. S. Bacquet, M. Zwingelstein-Colin, and F. Coudoux, "Reduced complexity H.264/AVC transrating based on frequency selectivity for high-definition streams" , *IEEE Trans. Consum. Electron.*, vol. 56, no. 4, pp. 2430–2437, Nov. 2010.
[11] I.-K. Kim, J. Min, T. Lee, W.-J. Han, and J. Park, "Block partitioning structure in the HEVC standard," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 22, no. 12, pp. 1697–1706, Dec. 2012.
[12] F. Bossen, B. Bross, K. Suhring, and D. Flynn, "HEVC complexity and implementation analysis," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 22, no. 12, pp. 1685–1696, Dec. 2012.
[13] L. Shen, Z. Zhang, and Z. Liu, "Effective CU size decision for HEVC intracoding," *IEEE Trans. Image Process.*, vol. 23, no. 10, pp. 4232–4241, Oct. 2014.
[14] L. Shen, Z. Zhang, and P. An, "Fast CU size decision and mode decision algorithm for HEVC intra coding," *IEEE Trans. Consum. Electron.*, vol. 59, no. 1, pp. 207–213, Feb. 2013.
[15] J. Xiong, H. Li, Q. Wu, and F. Meng, "A fast HEVC inter CU selection method based on pyramid motion divergence," *IEEE Trans. Multimedia*, vol. 16, no. 2, pp. 559–564, Feb. 2014.
[16] L. Shen, Z. Liu, X. Zhang, W. Zhao, and Z. Zhang, "An effective CU size decision method for HEVC encoders," *IEEE Trans. Multimedia*, vol. 15, no. 2, pp. 465–470, Feb. 2013.
[17] Z. Pan, S. Kwong, M.-T. Sun, and J. Lei, "Early MERGE mode decision based on motion estimation and hierarchical depth correlation for HEVC," *IEEE Trans. Broadcast.*, vol. 60, no. 2, pp. 405–412, Jun. 2014.
[18] X. Shen, and L. Yu, "CU splitting early termination based on weighted SVM," *EURASIP J. Image Video Process.*, vol. 2013, no. 4, Jan. 2013.
[19] G. Correa, P. Assuncao, L. Volcan Agostini, and L. da Silva Cruz, "Fast HEVC encoding decisions using data mining," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 25, no. 4, pp. 660–673, Apr. 2015.
[20] J. Lee, S. Kim, K. Lim, and S. Lee, "A fast CU size decision algorithm for HEVC," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 25, no. 3, pp. 411–421, Mar. 2015.
[21] S. Ahn, B. Lee, and M. Kim, "A novel fast CU encoding scheme based on spatiotemporal encoding parameters for HEVC inter coding," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 25, no. 3, pp. 422–435, Mar. 2015.
[22] L. Shen, Z. Zhang, and Z. Liu, "Adaptive inter-mode decision for HEVC jointly utilizing inter-level and spatiotemporal correlations," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 24, no. 10, pp. 1709–1722, Oct. 2014.
[23] D. Lefol, D. Bull, and N. Canagarajah, "Mode refinement algorithm for H.264 intra frame requantization," *Proc. IEEE Int. Symp. Circuits Syst.*, May 2006, pp. 4459–4462.
[24] D. Lefol and D. Bull, "Mode refinement algorithm for H.264 inter frame requantization," *Proc. IEEE Int. Conf. Image Process.*, Oct. 2006, pp. 845–848.
[25] T. Shanableh, E. Peixoto, and E. Izquierdo, "MPEG-2 to HEVC video transcoding with content-based modeling," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 23, no. 7, pp. 1191–1196, Jul. 2013.
[26] E. Peixoto, T. Shanableh, and E. Izquierdo, "H.264/AVC to HEVC video transcoder based on dynamic thresholding and content modeling," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 24, no. 1, pp. 99 112, Jan. 2014.
[27] J. De Praeter, J. De Cock, G. Van Wallendael, S. Van Leuven, P. Lambert, and R. Van de Walle, "Efficient transcoding for spatially misaligned compositions for HEVC," *Proc. IEEE Int. Conf. Image Process.*, Oct. 2014, pp. 2487–2491.
[28] G. Van Wallendael, J. De Cock, and R. Van de Walle, "Fast transcoding for video delivery by means of a control stream," *Proc. IEEE Int. Conf. Image Process.*, Sep. 2012, pp. 733–736.
[29] L. P. Van, J. De Cock, G. Van Wallendael, S. Van Leuven, R. Rodriguez-Sanchez, J. Martinez, P. Lambert, and R. Van de Walle, "Fast transrating for high efficiency video coding based on machine learning," *Proc. IEEE Int. Conf. Image Process.*, Sep. 2013, pp. 1573–1577.
[30] *High Efficiency Video Coding (HEVC) Test Model 7 Encoder Description* JCTVC-1002, May 2012

[31] *Common Test Conditions and Software Reference Configurations* JCTVC-I1100, May 2012.
[32] *Requirements of the Scalable Enhancement of HEVC*I. JTC1/SC29/WG11, Jul. 2012.
[33] I. H. Witten, and F. Eibe, *Data Mining: Practical Machine Learning Tools and Techniques*, San Mateo, CA, USA: Morgan Kaufmann, 2005.
[34] J. R. Quinlan, *Programs for Machine Learning*, San Mateo, CA, USA: Morgan Kaufmann, 1993.
[35] *Calculation of Average PSNR Differences Between RD-curves*Doc. VCEG-M33 of ITU-T VCEG, Apr. 2001.

Luong Pham Van (S'13) received the M.Sc. degree in electrical engineering from Sungkyunkwan University, Seoul, Korea, in 2011, and is currently working toward the Ph.D. degree at Ghent University, Ghent, Belgium.

His research interests include video compression, adaptation of video streams, transcoding, next generation video compression, and machine learning.

Johan De Praeter (S'14) received the M.Sc. degree in computer science engineering from Ghent University, Ghent, Belgium, in 2013, and is currently working toward the Ph.D. degree at Ghent University.

His research interests include video compression, high efficiency video coding, machine learning, and transcoding.

Glenn Van Wallendael received the M.Sc. degree in applied engineering from the University College of Antwerp, Antwerp, Belgium, in 2006, and the M.Sc. degree in engineering from Ghent University, Ghent, Belgium, in 2008.

He is currently a Post-Doctoral Researcher with the Multimedia Lab, Ghent University. His research interests include video compression, namely scalable video compression and transcoding.

Sebastiaan Van Leuven received the M.Sc. degree in applied engineering from the University College of Antwerp, Antwerp, Belgium, in 2006, and the M.Sc. and Ph.D. degree in computer science engineering from Ghent University, Ghent, Belgium, in 2008 and 2013, respectively.

In 2008 he joined the Multimedia Lab, Ghent University—iMinds. In 2011, he was a Visiting Researcher with the Universidad Castilla-La-Mancha, Albacete, Spain. Later in 2011, he was a Visiting Researcher with Florida Atlantic University, Boca Raton, FL, USA. Currently, he is a Post-Doctoral Researcher with the Multimedia Lab, Ghent University. His research interests include video coding, namely scalable video coding, transcoding, HEVC, and 3D video coding.

Jan De Cock (M'06) received the M.S. and Ph.D. degrees in engineering from Ghent University, Ghent, Belgium, in 2004 and 2009, respectively.

Since 2004, he has been with the Multimedia Lab, Ghent University—iMinds. His research interests include high-efficiency video coding and transcoding, scalable video coding, and multimedia applications.

Mr. De Cock was the recipient of a Postdoctoral Research Fellowship from the Flemish Agency for Innovation by Science and Technology (IWT) in 2010, and a Postdoctoral Research Fellowship from the Research Foundation Flanders (FWO) in 2012.

Rik Van de Walle (M'99) received the M.Sc. and Ph.D. degrees in engineering from Ghent University, Ghent, Belgium, in 1994 and 1998, respectively.

After a visiting scholarship at the University of Arizona, Tucson, AZ, USA, he returned to Ghent University, where he became Professor of Multimedia Systems and Applications, and Head of the Multimedia Lab. His current research interests include multimedia content delivery, presentation and archiving, coding and description of multimedia data, content adaptation, and interactive (mobile) multimedia applications.