LETTER

Fast Transform Unit Decision for HEVC

Jangbyung KANG[†], **Jin-Soo KIM**[†], **Jae-Gon KIM**[††], ***Nonmembers*, *and* Haechul CHOI**[†a)], ***Member***

SUMMARY For the High Efficiency Video Coding (HEVC) standard, a fast transform unit (TU) decision method is proposed. HEVC defines the TU representing a region sharing the same transformation, and it supports various transform sizes from 4×4 to 32×32 by using a quadtree of TUs. The various sizes of TUs can provide good coding efficiency, whereas it may dramatically increase encoding complexity. Assuming that a TU with highly compacted energy is unlikely to be split, the proposed method determines an appropriate TU size according to the position of the last non-zero transform coefficient. Experimental results show that this reduces encoding run time by 17.2% with a negligible coding loss of 0.78% BD-rate for the random-access scenario.

key words: *HEVC, transform unit, fast encoding*

1. Introduction

High Efficiency Video Coding (HEVC) has recently been developed by the Joint Collaborative Team on Video Coding (JCT-VC) of ITU-T VCEG and ISO/IEC MPEG [1]. One of the most significant changes from the previous video coding standards is the block partitioning structure. HEVC defines three different units: Coding Unit (CU), Prediction Unit (PU), and Transform Unit (TU), which represent respectively a region sharing the same prediction mode, a region sharing the same prediction information, and a region sharing the same transformation.

The ability to handle various block sizes of these units contributes high compression capability; however, because of the number of combinations of CU, PU, and TU, HEVC encoders inevitably require an excessive computational complexity. In order to reduce the computational complexity, Shen [2] proposed a fast CU size decision algorithm that is based on a Bayesian decision rule to avoid an exhaustive rate-distortion optimization (RDO) process for every possible CU size and its prediction mode. Choi [3] reduces the number of CU sizes to be searched by using coding tree pruning. Gweon [4] has proposed a fast PU size decision algorithm by using the signaling information coded_block_flag. These fast encoding methods described above have focused on early determination of CU and/or PU.

Manuscript received December 2, 2013.
Manuscript revised April 19, 2014.

[†]The authors are with the Department of Multimedia Engineering, Hanbat National University, Daejeon, Korea.
[††]The author is with School of Electronics, Telecommunication, and Computer Engineering, Korea Aerospace University, Gyeonggi-do, Korea.
a) E-mail: choihc@hanbat.ac.kr

DOI: 10.1587/transinf.E97.D.2205

In HEVC, TU is allowed to have various block sizes including 4×4, 8×8, 16×16, and 32×32. The HEVC reference model (HM) 12.0 [5] usually evaluates rate-distortion costs of all possible TU quadtree structures for each PU, which leads to high computational complexity. To reduce the high complexity of TU decision process, Choi [6] has proposed a TU quadtree pruning method. When the number of nonzero DCT coefficients is less than a threshold, a TU is not split into sub-blocks. Zhang [7] has proposed an adaptive residual quadtree (RQT) depth selection method. This method strictly sets the maximum inter RQT depth of 64×64 size CU to 1 and the maximum inter RQT depth of 8×8 size CU to 2. For other size CUs, TU split is determined according to the discriminant function obtained from the information of inter prediction residual including the current QP, variance of residual block, and variance of 4 sub-blocks of the residual block. This information should be extracted in an off-line way

In this letter, we exploit the relationship between TU split and the position of the last non-zero transform coefficient (LNTC) and propose a fast TU size decision method by pruning the quadtree of TUs at an early stage based on the position of the LNTC.

2. Fast Transform Unit Decision

In video coding techniques, a transform is used to represent a block of pixels with a small number of transform coefficients. This energy compaction scheme leads to compression with the help of quantization and entropy coding. The HEVC standard also adopts Discrete Cosine Transform (DCT) and Discrete Sine Transform (DST). Furthermore, it allows a residual block to be split into multiple TUs recursively with various transform block sizes, which achieves high energy compaction according to the characteristics of the residual block.

In principle, if energy of a TU is highly compacted as there are only a few low-frequency coefficients with no high frequency coefficients, the TU does not need to be split; whereas, if energy of the TU is insufficiently compacted as there are many high frequency coefficients, splitting the TU into 4 smaller TUs or higher depths has the potential for better coding efficiency. Consequently, the degree of energy compaction would be a suitable criterion for determining whether or not a TU should be split. In order to measure the degree of the energy compaction of a given transformed block, the proposed method uses the position of the

Table 1 Conditional probability that TUs are not split according to the position of the LNCT. p means the position of the LNTC.

Sequence	Conditional Probability ($\Pr(NotSplit \mid a \le p \le b)$) (%)		
	$0 \le p \le 5$	$6 \le p \le 20$	$21 \le p \le 40$
Traffic	89.75	41.13	35.45
Kimono	88.64	45.27	42.17
BasketallDrill	89.45	40.24	36.80
BQSquare	92.34	39.69	34.61
Average	**90.04**	**41.58**	**37.25**

LNTC. The position of the LNTC can be obtained simply and it rarely requires any additional complexity because it is a syntax element of HEVC.

The relationship between the TU split and the position of the LNTC is exploited statistically in the sense of RDO using HM 12.0. In Table 1, $\Pr(NotSpit \mid a \le p \le b)$ denotes the conditional probability that a TU is not split given the position of LNTC, p, $(a \le p \le b)$. Table 1 shows that TUs are more unlikely to be split in case the position of the LNTC is at a lower scan index. Therefore, it is verified that the optimal decision on the TU split has a high correlation with the position of the LNTC. Considering this correlation, the proposed method quickly makes a decision that a TU is not spilt if the LNTC is at a low frequency.

For each PU, an appropriate TU quadtree should be constructed. The TU quadtree has its root at the corresponding CU. That is, the maximum TU size is equal to the corresponding CU size. Starting from the maximum TU size, a TU can be recursively split into four smaller TUs. The minimum TU size, $MinTU_{size}$, is a larger value between 4×4 and the TU size associated with the maximum TU depth, $Depth_{MAX}$, as the following equation.

$$MinTU_{size} = \max\{TU(Depth_{MAX}), 4 \times 4\} \qquad (1)$$

In the proposed method, to determine the appropriate TU quadtree, the energy compaction of a given TU with depth i, TU_i, is tested by using (2).

$$Pos_{LNTC}(TU_i) \le T \qquad (2)$$

where $Pos_{LNTC}(TU_i)$ denotes the position of LNTC of TU_i according to the scanning method of HEVC. T is threshold value for the early TU split decision. When satisfying (1), the proposed method early terminates the TU decision process at the depth i without testing a sub-tree of higher TU depths.

The proposed fast TU decision algorithm is illustrated in Fig. 1. At each TU depth from which a particular TU size is inferred, TU size can be early determined as the current TU size in case the position of the LNTC of the current TU is lower than a predefined threshold. Otherwise, the TU is split into 4 smaller TUs and then the proposed algorithm is applied to each of the smaller TUs recursively. In case TU size has not been early determined and the current TU size is equal to the minimum TU size, all TU sizes are evaluated in the RDO process.

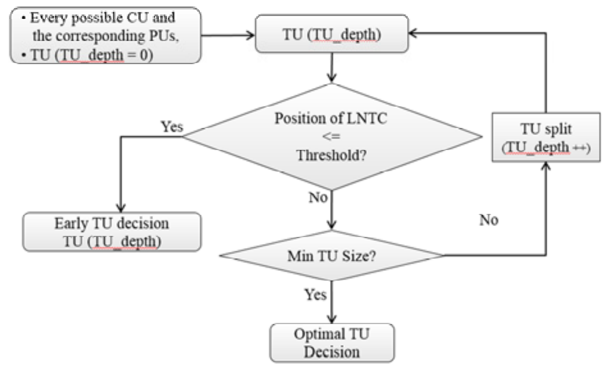

Fig. 1 The proposed fast TU decision algorithm.

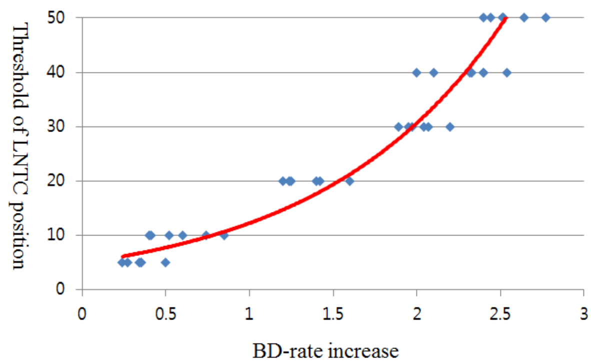

Fig. 2 BD-rate increase according to threshold value and its exponential fitting model.

Performance of the proposed TU decision method depends on the threshold value of the LNTC position. To provide a guide for the threshold value, various threshold values were tested. Figure 2 shows coding loss relative to the HM according to various threshold values. In the experiment of this figure, HEVC common test condition with Random_Access_Main for six test sequences including class A to class E [8] was used and quantization parameter was set to 32. It is observed that the BD-rate monotonically increases as the threshold value increase. The BD-rate increase is an inevitable cost for the purpose of reducing computational complexity. The experimental results can be modeled by an exponential function as follows:

$$T = w_1 \times e^{w_2 \times BDR} \qquad (3)$$

where T is the threshold value for the TU split decision and BDR is the amount of the BD-rate increase. w_1 and w_2 are parameters of the exponential function to fit the experimental result of Fig. 2 to the model as closely as possible. For the experimental result, the values of w_1 and w_2 are 3.233 and 1.12 respectively, which minimizes the sum of square error. By using the exponential model, an appropriate threshold value can be derived easily for an acceptable coding loss as represented by the BD-rate. In this way, the proposed method can provide a useful feature for controlling the tradeoff between computational complexity and coding efficiency.

3. Experimental Results

The proposed method is implemented on the top of HM 12.0. In experiments, the common test sequences of HEVC including classes A, B, C, and D are used, and encoder control follows the HEVC common test condition with Random_Access_Main and Low_Delay_P_Main configurations [8]. The HM 12.0 adopts fast encoding schemes to determine the CU size and the corresponding PUs [3], [4] that can reduce encoding time by about 50%. This fast encoding scheme is always turned on in all experiments to evaluate the proposed method. To compare the performance of computational complexity, encoding time is measured with time saving (TS) as follows:

$$TS(\%) = (TS_{ref} - TS_{prop})/TS_{ref} \times 100 \tag{4}$$

where TS_{ref} and TS_{prop} are encoding times of HM 12.0 and the proposed method, respectively.

Table 2 and Table 3 show experimental results of Zhang's method [7] and the proposed method relative to HM 12.0. As described in Introduction, Zhang's method strictly sets the TU size of 64 × 64 CU to 32 × 32, and the TU sizes of 32 × 32 PU and 16 × 16 PU are determined by the discriminant function extracted in an off-line way. In these experiments, parameters for the discriminant function are trained using HM 12.0 for sequences Traffic, Kimono, BasketballDrill, and BasketballPass. On the other hand, the proposed method directly determines TU size according to the position of the LNTC. In these experiments, the threshold value of the proposed method is set to 5. As shown in Table 2, the proposed method achieves an average 17.2% reduction in encoding time relative to HM 12.0 for the Random_Access_Main case, while the coding loss is only an average 0.78% in the BD-rate. For the Low_Delay_P_Main case, the encoding time is reduced by about 22.76%, while the coding loss is only 1.02% in the BD-rate. The Zhang's method reduces the encoding times by about 15.6% and 18.5% for the Random_Access_Main case and the Low_Delay_P_Main case, respectively. Compared with the Zhang's method, the proposed method reduces encoding time 2 to 4% more, while the coding loss is almost same. Furthermore, the proposed method does not need a training sequence set and an off-line calculation. In contrast, the Zhang's method should derive the discriminant function using a training sequence set in advance.

The performance of TU depth = 1 case was also tested under the same experimental condition of Table 2 and Table 3. This case has average coding losses of 0.9% and 1.0% in BD-rate relative to HM 12.0 under the Random_Access_Main and Low_Delay_P_Main configurations, respectively. Encoding times are reduced by 15.4% and 18.7% under the configurations, respectively. Compared with the TU depth = 1 case, the proposed method reduces encoding time 1.8% to 4.06% more. Furthermore, the coding loss is less than or equal to the TU depth = 1 case.

Compared with HM 12.0, these results verify that the proposed method is able to reduce computational complexity significantly with a negligible coding loss. The coding loss for other thresholds of the LNTC is shown in Fig. 2. Note that the encoding time reduction is achieved although the fast encoding methods of HM 12.0 are turned on in the performance comparison. This means that the proposed method obtains additional complexity reduction and works well together with other fast CU and PU decision methods.

To analyze the effect and the accuracy of the proposed method, sequences Nebuta and StreamLocomotive are tested. In Table 2, these sequences have the lease and most coding losses, respectively. In Table 4, when the threshold of the position of the LNTC is set to 5, the ratio of TUs satisfying (2) to all TUs is listed in the third column. All TUs satisfying (2) are decided not to be split in the proposed method. This ratio also implies the amount

Table 2 Performance under Random_Access_Main configuration.

Sequence		Zhang [7]		Proposed method	
		BD-rate (%)	TS (%)	BD-rate (%)	TS (%)
Class A	Traffic	0.3	12	0.3	15
	PeopleOnStreet	0.7	14	0.8	18
	Nebuta	0.2	18	0.2	20
	SteamLocomotive	2.0	12	2.2	14
Class B	Kimono	1.0	16	1.1	16
	ParkScene	0.8	15	0.7	15
	Cactus	0.4	14	0.5	18
	BasketballDrive	0.8	12	0.6	15
	BQTerrace	0.8	16	0.8	20
Class C	BasketballDrill	0.3	14	0.4	16
	BQMall	0.5	13	0.5	16
	PartyScene	0.2	16	0.3	18
	RaceHorsesC	1.2	19	1.1	18
Class D	BasketballPass	1.1	18	1.0	16
	BQsquare	1.3	18	1.2	20
	BlowingBubbles	0.4	20	0.7	20
	RaceHorses	1.2	19	0.9	18
Average		**0.77**	**15.6**	**0.78**	**17.2**

Table 3 Performance under Low_Delay_P_Main configuration.

Sequence		Zhang [7]		Proposed method	
		BD-rate (%)	TS (%)	BD-rate (%)	TS (%)
ClassA	Traffic	0.4	16	0.4	24
	PeopleOnStreet	0.6	14	0.7	21
	Nebuta	0.3	23	0.3	21
	SteamLocomotive	0.4	14	0.4	19
Class B	Kimono	1.4	18	1.0	20
	ParkScene	1.4	20	1.0	23
	Cactus	0.2	16	1.0	24
	BasketballDrive	0.6	14	1.2	19
	BQTerrace	1.5	23	0.8	27
Class C	BasketballDrill	0.4	16	1.4	22
	BQMall	0.7	16	0.8	23
	PartyScene	0.7	19	1.5	27
	RaceHorsesC	1.5	19	1.7	20
Class D	BasketballPass	1.6	19	1.2	19
	BQsquare	1.8	24	1.1	30
	BlowingBubbles	1.9	25	1.4	28
	RaceHorses	1.9	20	1.5	20
Average		**1.01**	**18.5**	**1.02**	**22.76**

Table 4 Effect and accuracy of the proposed method.

Sequence	QP	TUs not split by proposed method (A) (%)	TUs not split in RDO sense (B) (%)	Wrong decision (A-B) (%)
Nebuta	22	13.40	13.20	0.20
	27	35.42	35.21	0.21
	32	65.03	64.70	0.33
	37	83.25	81.40	1.85
SteamLoco-motive	22	77.58	77.55	0.03
	27	90.96	87.66	3.30
	32	93.96	90.04	3.92
	37	96.05	91.84	4.21

of TUs influenced by the proposed method. It is clear that the position of the LNTC decreases as the quantization parameter increases because more transform coefficients are quantized to zeros. Thus, the effect of the proposed method also increases as the quantization parameter increases. The ratio of TUs that satisfy (2) and are not split in the sense of RDO by the HM 12.0 is listed in the fourth column. The difference of the two ratios indicates the ratio of TUs that are split in the sense of RDO although the position of the LNTC is smaller than and equal to the threshold. Thus, the amount of wrong decision made by the proposed method can be estimated from this difference. In Table 4, the amount of the wrong decision increases slightly as quantization parameter increases. As described above, this is because more TUs are affected by the proposed method under the condition of higher quantization parameter. If the position of the LNTC of a TU is within a predefined low frequency but some smaller TUs generated by splitting the TU have no significant transform coefficient, splitting the TU may be better because the number of bits required to represent transform coefficients can be dramatically saved using syntax element *coded_sub_block_flag*. Thus, the wrong decision can happen although the position of the LNTC of a TU is within a predefined low frequency. However, as shown in Table 4, the amount of the wrong decision is still very low when quantization parameter is high. The proposed method thereby maintains almost coding efficiency as shown in Tables 2 and 3. In Table 2, the coding loss for sequence Nebuta is lower than that for sequence SteamLocomotiv. As expected, the wrong decision ratio for sequence Nebuta is also lower than that for sequence SteamLocomotiv.

4. Conclusion

For the purpose of reducing the computational complexity of HEVC encoders, the proposed method terminates at an early stage the search for possible smaller TU sizes when the position of the LNTC of a TU is within a predefined low frequency. As shown in the experimental results, an average 17.2% reduction of encoding time is achieved with a negligible coding loss of a BD-rate increase of 0.78% for Random_Access_Main case. Moreover, the proposed method provides a good trade-off model between computational complexity and coding efficiency that is represented as an exponential function. The proposed method can contribute to the design of a fast HEVC encoder.

Acknowledgments

This work was supported by the National Research Foundation of Korea (NRF) Grant funded by the Korean Government (MSIP) (NRF-2013R1A1A1010344) and the Ministry of Education (MOE) and NRF through the Human Resource Training Project for Regional Innovation (2012H1B8A2025982).

References

[1] Joint Collaborative Team on Video Coding (JCT-VC), "High Efficiency Video Coding (HEVC) text specification draft 10 (for FDIS & Consent)," JCTVC-L1003, Geneva, Jan. 2013.

[2] X. Shen, L. Yu, and J. Chen, "Fast coding unit size selection for HEVC based on Bayesian decision rule," Proc. Picture Coding Symposium, pp.453–456, May 2012.

[3] K. Choi, S.H. Park, and E.S. Jang, "Coding tree pruning based CU early termination," JCTVC-F092, Torino, Italy, July 2011.

[4] R. Gweon and Y.-L. Lee, "Early termination of CU encoding to reduce HEVC complexity," JCTVC-F405, Torino, Italy, July 2011.

[5] High Efficiency Video Coding Test Model Software 12.0, https://hevc.hhi.fraunhofer.de/svn/svn_HEVCSoftware.

[6] K. Choi and E.S. Jang, "Early TU decision method for fast video encoding in high efficiency video coding," Electron. Lett., vol.48, pp.689–691, June 2012.

[7] Y. Zhang and M. Zhao, "An adaptive RQT mode selection algorithm for HEVC," Proc. Image and Signal Processing, pp.173–177, Oct. 2012.

[8] JCT-VC, "Common HM test conditions and software reference configurations," JCTVC-I1100, Geneva, CH, May 2012.