Accepted in IEEE Int. Conference on Image Processing (ICIP 2015), 2015

FAST H.264 TO HEVC TRANSCODER BASED ON POST-ORDER TRAVERSAL OF QUADTREE STRUCTURE

Jean-François Franche, Stéphane Coulombe

Department of Software and IT Engineering
École de technologie supérieure, Université du Québec
Montréal, Québec, Canada

ABSTRACT

In this paper, we propose a fast mode decision framework and a fast motion estimation algorithm for H.264 to High Efficiency Video Coding (HEVC) transcoding. The fast mode decision framework employs a post-order (bottom-up) traversal of the coding tree unit (CTU) quadtree. Based on this traversal and H.264 information, several strategies are proposed to reduce HEVC modes to be tested and a rate-distortion (RD) cost prediction model is used to terminate the processing of a tested mode early. The proposed fast motion estimation algorithm selects the best candidate from a list of H.264 motion vectors (MVs) and previously encoded HEVC MVs. Compared to a full re-encoding, experimental results show that the proposed solution achieves speed-ups of up to 12.75x, for an average BD-Rate of 3.28%.

***Index Terms*—** H.264/AVC, H.265/HEVC, video transcoding, fast mode decision, motion estimation

1. INTRODUCTION

In 2013, the Joint Collaborative Team on Video Coding (JCT-VC) has completed H.265/High Efficiency Video Coding (HEVC), the most recent video compression standard. Compared to its predecessor, H.264/AVC, HEVC can save approximately 50% of the bitrate for similar video quality [1]. To take advantage of HEVC coding efficiency and to ensure systems interoperability, several H.264 sequences must be transcoded to HEVC.

The simplest video transcoding approach, called cascade pixel-domain transcoding (CPDT) [2], decodes the input sequence entirely and re-encodes the pixel data in the output format. This approach achieves high coding efficiency and offers great flexibility on video encoding parameters. However, it is very complex computationally. To reduce this complexity, several approaches reuse extracted information - such as modes, motion information and encoded residuals - from the incoming bitstream to speed up the encoding process. For inter frames, these approaches typically focus on fast mode decision and fast motion estimation.

Several fast mode decision approaches have been proposed in the literature [3–12]. For example, Fang et al. use H.264 modes, encoded residual and variance of motion vectors (MVs) to skip some prediction unit (PU) modes [3]. Jiang et al. use the number of H.264 bits to decide the searching depth range of a coding tree unit (CTU) and determine the coding unit (CU) and PU modes to be tested by an H.264 MV clustering method [4]. To achieve better performances, Peixoto et al. exploit HEVC information in addition to extracted H.264 information [5]. Their proposed approach uses a linear discriminant model based on H.264 features to determine if a CU must be split. Based on a statistical model, the rate distortion (RD) cost of tested HEVC modes allows additional modes to be disabled.

To reduce the computational complexity of motion estimation, Peixoto et al. select the best H.264 MV in the current CU and refine this MV at half and quarter pixel precisions [5]. Shen et al. select the H.264's dominant MV as a starting point, and refine this MV with a search range of 4 pixels [10]. Other authors propose to use a dynamic search range [3, 8]. For example, Zong et al. use the advanced motion vector prediction (AMVP) as a starting point, and define the search range as the distance between this point and the H.264's dominant MV [8].

In this paper, a fast mode decision framework and a fast motion estimation algorithm are proposed. The fast mode decision framework is distinguished from others by a post-order (bottom-up) traversal of the CTU quadtree and the use of a prediction model to avoid evaluations of the RD cost function. Based on this traversal and H.264 information, several strategies are proposed to reduce the HEVC modes to be tested. Furthermore, an RD cost prediction model is used to terminate the processing of a tested mode early. The proposed fast motion estimation algorithm selects the best MV from a list created at the CTU level and composed of H.264 MVs and previously encoded HEVC MVs. Compared to related works, this method does not require motion refinement. The

This work was funded by Vantrix Corporation and by the Natural Sciences and Engineering Research Council of Canada under the Collaborative Research and Development Program (NSERC-CRD 428942-11). Emails: jean-francois.franche.1@ens.etsmtl.ca, stephane.coulombe@etsmtl.ca

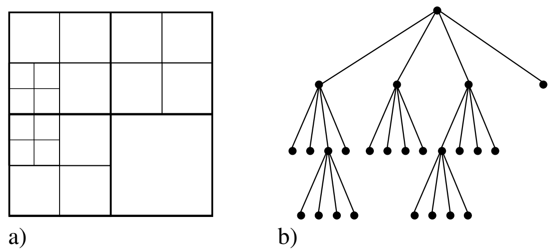

Fig. 1. Example of a) CTU partitioning with its b) quadtree representation.

proposed transcoder leads to better speed-ups than those of related works which preserve the visual quality.

The rest of this paper is organized as follows. Section 2 presents the background on CTU and the rate-distortion optimization (RDO) process of the reference encoder, HM 12.1. The proposed fast mode decision framework and fast motion algorithm are respectively described in Sections 3 and 4. The experimental setup and results are presented in Section 5. Section 6 concludes the paper and presents future work.

2. HEVC PARTITIONING AND MODE SELECTION

In HEVC, a CTU is formed by a quadtree structure having a maximum of 4 depth levels (from 0 to 3), and representing a region of up to 64×64 pixels [13]. Each node of a CTU is associated with a CU noted $C_{i,j}$, where j is the jth CU at depth level i. The CU $C_{i,j}$ can be split into 4 sub-CUs of the same size. Indices of these sub-CUs are $(i+1, 4j+k)$, with $k = 0..3$. An example of a CTU quadtree and its corresponding partitioning structure is shown in Fig. 1.

The prediction model of each CU is given by a PU. The prediction mode of a PU can be intra, inter or skip/merge. For inter PUs, 4 symmetric motion partition modes (2N×2N, N×2N, 2N×N, and N×N) and 4 asymmetric motion partition (AMP) modes (2N×nD, 2N×nU, nL×2N, and nR×2N) are available. Intra PUs support only 2 partition modes (2N×2N and N×N), and skip/merge PU, only one (2N×2N).

In the HM 12.1 reference encoder, the coding efficiency of a mode is evaluated by an RD cost function defined as:

$$J_{\text{RD}} = (\text{SSE}_{\text{Luma}} + 0.57\,\text{SSE}_{\text{Chroma}}) + \lambda_{\text{mode}} \cdot B_{\text{mode}}, \quad (1)$$

where SSE is the sum of square errors between the original input image block and the reconstructed block, B_{mode} is the number of bits to encode the current mode, and λ_{mode} is the lambda value depending on the quantization parameter (QP).

To determine the best mode, the RDO process performs a pre-order traversal on the CTU. Indices in the nodes of Fig. 2.(a) show the order in which CUs are visited by this traversal method (for a CTU with 3 depth levels only). When a CU is visited, its inter, intra, skip and merge modes are fully encoded. The mode with the smallest J_{RD} value is selected as the best CU mode. If the maximum depth is not reached, the current CU is split into 4 sub-CUs. These nodes are then recursively visited. When all the descendants of the current CU have been visited, J_{RD} of this CU is compared with the combined J_{RD} of these 4 sub-CUs to decide whether or not the CU must be split. At the end of the process, the best mode and CTU partitioning are obtained.

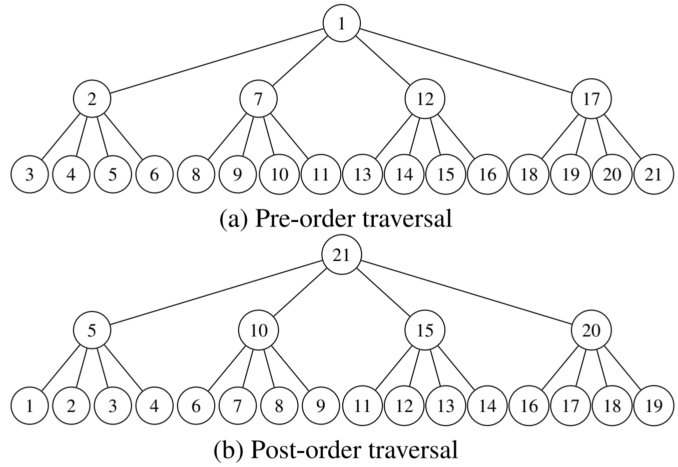

Fig. 2. Comparison between a) HM CTU traversal and b) proposed CTU traversal.

3. PROPOSED FAST MODE DECISION FRAMEWORK

In a pre-order traversal, the early termination of CU splitting is a complex task because many combinations of sub-CUs and PUs are possible, and any of them may reduce the best RD cost. Moreover, in this type of traversal, a sub-CU cannot be compared directly to its parent to be early terminated, because their sizes are different.

To address these issues, the proposed fast mode decision framework is based on a post-order traversal of the CTU quadtree. In this type of traversal, descendants of the current CU are first visited recursively, as shown in Fig. 2.(b). Then, the PU modes of the current CU are processed. This traversal changes the CU splitting problem of pre-order traversal into a sub-CUs merging problem. This is an easier problem to handle, because the current best combination of sub-CUs and PUs is in competition with only one CU, and only modes of the same size must be compared.

The proposed framework is shown in Algorithm 1. The first part (lines 3-7) of this algorithm processes all descendants of the current CU recursively and the second part (lines 8-15) processes the PU modes. The SPLIT, MERGE and MODES functions reuse either, information extracted from H.264 or information on HEVC processed modes to reduce the modes tested as explained in Sec. 3.1. To further improve speed-up, an RD cost prediction model allows early termination of mode processing as described in Sec. 3.2.

Algorithm 1 Proposed fast mode decision framework

1: **function** PROCESSCU($C_{i,j}$)
2: $J_{\text{RD}}[C_{i,j}] \leftarrow J_{\text{PM}}[C_{i,j}] \leftarrow \infty$
3: **if** SPLIT($C_{i,j}$) **then**
4: **for** $k = 0..3$ **do**
5: PROCESSCU($C_{i+1,4j+k}$)
6: $J_{\text{RD}}[C_{i,j}] \leftarrow \lambda_{\text{pred}} \cdot B_{\text{split}} + \sum_{k=0}^{3} J_{\text{RD}}[C_{i+1,4j+k}]$
7: $J_{\text{PM}}[C_{i,j}] \leftarrow \lambda_{\text{mode}} \cdot B_{\text{split}} + \sum_{k=0}^{3} J_{\text{PM}}[C_{i+1,4j+k}]$
8: **if** MERGE($C_{i,j}$) **then**
9: **for each** m **in** MODES($C_{i,j}$) **do**
10: $J_{\text{PM}}[m] \leftarrow$ PMCOST($C_{i,j}$,m)
11: **if** $(J_{\text{PM}}[m] - J_{\text{PM}}[C_{i,j}]) \geq T$ **then**
12: $J_{\text{RD}}[m] \leftarrow$ RDCOST($C_{i,j}$,m)
13: **if** $J_{\text{RD}}[m] < J_{\text{RD}}[C_{i,j}]$ **then**
14: $J_{\text{RD}}[C_{i,j}] \leftarrow J_{\text{RD}}[m]$
15: $J_{\text{PM}}[C_{i,j}] \leftarrow J_{\text{PM}}[m]$

3.1. Modes reduction

The first step (line 3-5) of Algorithm 1 consists in recursively splitting the current CU at its deepest level. Considering that the HEVC partitioning is rarely finer than H.264's partitioning, the SPLIT function performs a direct mapping of H.264 partitioning to HEVC maximal depth level. Thus, CUs at depth level 0 and 1 are always split since their size is larger than a macroblock (MB). At depth level 2, the CU is only split if the corresponding H.264 MB contains partitions of 8×8 pixels or less.

When all descendants of the current CU are processed, the MERGE function (line 8) analyzes the current best mode. If this mode contains a CU deeper than $i + 1$, the region is considered complex, merge is deactivated and all modes of the current CU are skipped. This means that merging at this and higher levels is not desired and we can stop trying to merge on this branch.

If merge is activated, the MODES function (line 9) returns the PU modes to be tested. Due to similar prediction mode (inter or intra) between H.264 and HEVC, inter modes are activated when a CU region contains at least an inter H.264 MB. Similarly, intra modes are activated when a CU region contains at least an intra MB. Since AMP modes are time-consuming and have little influence on the coding efficiency [4, 10], these modes are always disabled. For opposite reasons, skip and merge modes are always activated. Finally, the activated modes are processed in the following order: inter modes, skip / merge modes and intra modes, from finer to coarser partitions.

3.2. Early termination of mode processing

The RD cost computation of a mode, performed by RDCOST (line 12), is a complex task that involves a complete encoding of this mode. To reduce complexity of this process, the PMCOST function (line 10) estimates the RD cost by an prediction model cost defined as:

$$J_{\text{PM}} = (\text{SATD}_{\text{Luma}}) + \lambda_{\text{pred}} \cdot B_{\text{pred}}, \qquad (2)$$

where SATD is the sum of absolute Hadamard transformed coefficients of the prediction error, λ_{pred} is equal to $\sqrt{\lambda_{\text{mode}}}$ and B_{pred} is the number of bits to encode prediction information (in lines 6 and 7, B_{split} is the number of bits to encode the split flag). For inter modes, J_{PM} is computed during the motion estimation process. For intra modes, J_{PM} is computed for every candidate and the best value is preserved. Both of them skip the complex RD cost evaluation task which includes transform, quantization and entropy encoding.

If the difference between $J_{\text{PM}}[C_{i,j}]$, the current best cost for the CU $C_{i,j}$, and $J_{\text{PM}}[m]$, cost for the mode m, is lower than a threshold T (line 11), the mode processing is early terminated. The threshold T controls the trade-off between coding efficiency and computational complexity of the transcoder. For experiments, it is empirically set to $3\times\lambda_{\text{pred}}$.

4. PROPOSED FAST MOTION ESTIMATION ALGORITHM

In our experiments, we observed that the best MV found by the HM's fast motion estimation algorithm is often an H.264 MV located in the current PU region. However, since MVs tend to propagate spatially, it may correspond to an H.264 or an HEVC MV located in the neighborhood of the current PU region. Moreover, considering that the HEVC motion estimation process uses temporal MV candidates, it may also be an HEVC MV co-located in the reference frame. Finally, the MV found may correspond to neither of these cases.

Based on these observations, the proposed fast motion estimation algorithm creates an MV candidates list for the current CTU. This list is composed of m H.264 and HEVC MVs, as shown in Fig. 3. H.264 MVs are co-located in the CTU region in the current H.264 frame. MVs in the neighborhood are also considered. HEVC MVs are either co-located in the CTU region in the HEVC reference frame or located in the processed neighborhood of the CTU in the current HEVC frame. All duplicated MVs are removed from the list. Each PU mode selects the best MV in this list based on the RD cost prediction model defined in Eq. 2. Unlike related methods, no motion refinement is performed since the proposed list is accurate.

Considering that all the PU modes of the current CTU test the same MVs, error prediction of each MV candidate is pre-computed and stored in an $m\times(\text{CTU}_{\text{size}}/4)\times(\text{CTU}_{\text{size}}/4)$ matrix, denoted $\mathbf{E}$, and where CTU_{size} is the CTU size (normally 64). Each element of $\mathbf{E}$ represents error prediction from a region of 4×4 pixels, and is computed as:

$$e_{k,i,j} = \text{SATD}_{4\times4}(\mathbf{R}_{k,4i,4j}), \qquad (3)$$

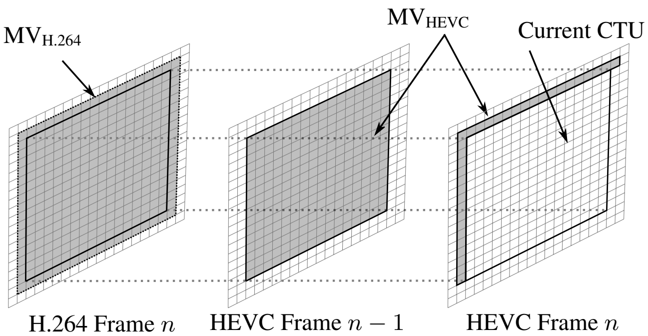

Fig. 3. Regions (in grey) of motion vector candidates.

with k=0..$(m-1)$ and i,j = 0..15; where $\mathbf{R}$ is a matrix of $m\times$CTU$_{size}$ $\times$ CTU$_{size}$ and contains differences between the processed CTU region and predicted samples of each m MV candidate, $\mathrm{SATD}_{4\times4}$ is the sum of absolute Hadamard transformed coefficients for a 4×4 region, noted $\mathbf{R}_{k,4i,4j}$. In HM, $\mathrm{SATD}_{4\times4}$ is only used for 4×8 and 8×4 PUs. For other sizes, $\mathrm{SATD}_{8\times8}$ is employed. According to our experimental results, $\mathrm{SATD}_{4\times4}$ is sufficiently accurate for our algorithm.

For a PU of $N\times M$ pixels, located at position $(x,\ y)$ relative to the CTU's upper left corner, the prediction error of MV k is computed as:

$$\mathrm{ER}_{PU}(k,x,y,N,M) = \sum_{i=(x/4)}^{(x+N)/4-1} \sum_{j=(y/4)}^{(y+M)/4-1} e_{k,i,j} \quad (4)$$

5. EXPERIMENTAL RESULTS

To evaluate the proposed transcoder, experiments with standard test sequences of HEVC were conducted. The proposed transcoding approach is compared to the CPDT transcoding approach, described in Sec. 1. The test sequences were encoded and transcoded with QP 22, 27, 32 and 37. An IPPP coding structure and one reference frame were employed. The H.264 bitstream was generated with a JM 18.2 baseline profile. The fast full search motion algorithm was employed with a search range of [-64,64]. The CPDT and the proposed transcoding approach implementations were based on HM 12.1. The HM fast search algorithm, with a search range of [-64,64], was employed by the CPDT transcoder. For all tests, HEVC sequences were evaluated for a low delay setting and low complexity configuration. Performance was measured using the Bjøntegaard Delta-Rate (BD-Rate) [14] and speed-up factor. Speed-up is based HEVC encoding time (H.264 decoding is not considered, but would have little impact on the results).

Table 1 shows the proposed approach's performance relative to the CDPT transcoder. Two sets were tested : the fully proposed approach (Full) and the proposed approach with only fast motion estimation (FME) activated. For some sequences, the FME method decreases the BD-Rate. These results are explained by the fact that the proposed algorithm performs its processing directly in quarter pixel precision and H.264 uses the full fast search. Another important observation is that the FME method gets better speed-up for higher QPs. These are expected results, since the relative complexity of the motion estimation increases with the QP.

When the fully proposed approach is used, speed-up increases dramatically and the coding efficiency loss is still reasonable (the BD-Rate rarely increases beyond 5%, and the average is 3.28). Generally, the proposed approach gets better speed-ups for sequences with complex motion activities, such as RaceHorses and Keiba, and worse speed-ups for sequences with simple motion activities and high intra modes proportion, like Mobisode2 and Flowervase. Finally, the proposed approach achieves an average speed-up (7.89x) better than related approaches, such as Chen et al [11] (1.79x), Jiang et al. [4] (2.00x) and Peixoto et al. [5] (3.83x), for instance.

Table 1. Speed-up and BD-Rate of the proposed H.264 to HEVC transcoder for Full and FME methods.

Sequence	Method	Speed-up (by QP)				BD-Rate (%)
		22	27	32	37	
RaceHorses	FME	1.04	1.07	1.13	1.18	-0.69
416×240×30	Full	10.01	8.60	7.66	6.88	2.31
BasketballPass	FME	1.09	1.13	1.17	1.27	0.21
416×240×50	Full	7.54	7.12	6.40	5.73	2.88
BQSquare	FME	1.05	1.07	1.15	1.28	0.21
416×240×60	Full	11.65	9.95	6.99	5.26	1.16
Flowervase	FME	1.20	1.26	1.33	1.33	0.24
416×240×30	Full	6.85	5.71	5.22	5.00	1.94
Keiba	FME	1.07	1.18	1.24	1.33	-1.01
416×240×30	Full	9.04	7.23	6.42	6.06	2.61
Mobisode2	FME	1.25	1.30	1.31	1.33	-0.92
416×240×30	Full	6.39	6.04	5.65	5.40	3.21
BasketballDrill	FME	1.18	1.24	1.33	1.40	-1.38
832×480×50	Full	10.03	8.82	7.97	7.48	5.27
BQMall	FME	1.17	1.23	1.32	1.39	-0.75
832×480×60	Full	11.39	9.74	8.13	7.35	4.76
Flowervase	FME	1.20	1.30	1.42	1.48	-1.19
832×480×30	Full	10.51	8.92	7.41	6.59	2.61
Keiba	FME	1.23	1.30	1.40	1.49	-1.03
832×480×30	Full	8.20	7.73	7.56	7.45	5.22
PartyScene	FME	1.12	1.17	1.26	1.33	-0.51
832×480×50	Full	12.75	11.04	8.43	7.0	2.57
RaceHorses	FME	1.14	1.19	1.29	1.39	-0.55
832×480×30	Full	9.68	9.36	8.53	7.90	4.73

6. CONCLUSION

In this paper, we presented a fast H.264 to HEVC transcoder. An early termination method was proposed to terminate the processing of a PU. To increase usage of this early termination method, CTU uses post-order traversal. A fast motion estimation algorithm based on a MV candidates list was also proposed. The proposed transcoder achieves high speed-up with a low coding efficiency reduction. To improve the proposed approach, we plan to study fast intra prediction methods and improve the proposed models.

7. REFERENCES

[1] J. Ohm, G.J Sullivan, H. Schwarz, T.K. Tan, and T. Wiegand, "Comparison of the coding efficiency of video coding standards—including high efficiency video coding (HEVC)," *Circuits and Systems for Video Technology, IEEE Transactions on*, vol. 22, no. 12, pp. 1669–1684, 2012.

[2] A. Vetro, C. Christopoulos, and H. Sun, "Video transcoding architectures and techniques: an overview," *Signal Processing Magazine, IEEE*, vol. 20, no. 2, pp. 18–29, 2003.

[3] J.T. Fang, Z.Y. Chen, T.L. Liao, and P.C. Chang, "A fast PU mode decision algorithm for H.264/AVC to HEVC transcoding," *Computer Science*, 2014.

[4] W. Jiang, Y. Chen, and X. Tian, "Fast transcoding from H.264 to HEVC based on region feature analysis," *Multimedia Tools and Applications*, pp. 1–22, 2013.

[5] E. Peixoto, B. Macchiavello, E.M. Hung, and R.L. de Queiroz, "A fast HEVC transcoder based on content modeling and early termination," *21st IEEE International Conference on Image Processing (ICIP 2014)*, pp. 1–5, 2014.

[6] D. Zhang, B. Li, J. Xu, and H. Li, "Fast transcoding from H.264/AVC to high efficiency video coding," in *Multimedia and Expo (ICME), 2012 IEEE International Conference on*. IEEE, 2012, pp. 651–656.

[7] W. Jiang and Y.W. Chen, "Low-complexity transcoding from H.264 to HEVC based on motion vector clustering," *Electronics Letters*, vol. 49, no. 19, pp. 1224–1226, 2013.

[8] C. Zong-Yi, Tseng C.T., and C. Pao-Chi, "Fast inter prediction for H.264 to HEVC transcoding," in *3rd International Conference on Multimedia Technology (ICMT-13)*. Atlantis Press, 2013.

[9] R. Luo, R. Xie, and L. Zhang, "Fast AVS to HEVC transcoding based on ROI detection using visual characteristics," in *Broadband Multimedia Systems and Broadcasting (BMSB), 2014 IEEE International Symposium on*. IEEE, 2014, pp. 1–6.

[10] T. Shen, Y. Lu, Z. Wen, L. Zou, Y. Chen, and J. Wen, "Ultra fast H.264/AVC to HEVC transcoder," in *Data Compression Conference (DCC), 2013*. IEEE, 2013, pp. 241–250.

[11] Z.-Y. Chen, J.-T. Fang, T.-L. Liao, and P.-C. Chang, "Efficient PU mode decision and motion estimation for H.264/AVC to HEVC transcoder," *Signal*, 2014.

[12] P. Xing, Y. Tian, X. Zhang, Y. Wang, and T. Huang, "A coding unit classification based AVC-to-HEVC transcoding with background modeling for surveillance videos," in *Visual Communications and Image Processing (VCIP), 2013*. IEEE, 2013, pp. 1–6.

[13] G.J. Sullivan, J. Ohm, W.J. Han, and T. Wiegand, "Overview of the high efficiency video coding (HEVC) standard," *Circuits and Systems for Video Technology, IEEE Transactions on*, vol. 22, no. 12, pp. 1649–1668, 2012.

[14] G. Bjøntegaard, "document VCEG-M33: Calculation of average PSNR differences between RD-curves," in *ITU-T VCEG Meeting, Austin, Texas, USA, Tech. Rep*, 2001.