Fast quadtree level decision algorithm for H.264/HEVC transcoder

A. J. Diaz-Honrubia, J. L. Martinez, J. M. Puerta, J. A. Gamez, J. De Cock, and P. Cuenca

In: 2014 IEEE International Conference on Image Processing (ICIP) , 2497-2501, 2014.

FAST QUADTREE LEVEL DECISION ALGORITHM FOR H.264/HEVC TRANSCODER

A. J. Díaz-Honrubia[1], *J. L. Martínez*[1], *J. M. Puerta*[1], *J. A. Gámez*[1], *J. De Cock*[2] *and P. Cuenca*[1]

[1] Albacete Research Institute of Informatics (I3A), University of Castilla-La Mancha, Spain
Email: {Antonio.DHonrubia, JoseLuis.Martinez, Jose.Puerta, Jose.Gamez, Pedro.Cuenca}@uclm.es

[2] iMinds, ELIS, Multimedia Lab, Ghent University, Ledeberg-Ghent, Belgium
Email: jan.decock@ugent.be

ABSTRACT

The High Efficiency Video Coding (HEVC) was developed by the Joint Collaborative Team on Video Coding (JCT-VC) to replace the current H.264/AVC standard which has been widely adopted in the last years. Therefore, there is a lot of legacy content encoded with H.264/AVC and an efficient conversion to HEVC is needed. This paper, presents a Fast Quadtree Level Decision (FQLD) algorithm that greatly reduces the complexity of the transcoding process between H.264/AVC and HEVC. The proposal tries to exploit the information gathered at the H.264/AVC decoder to make decisions on Coding Units (CU) splitting in HEVC using a Naïve-Bayes (NB) probabilistic classifier. Experimental results show that the proposed transcoder can achieve a good tradeoff between coding efficiency and complexity.

Index Terms— H.264/AVC, HEVC, Transcoding, CTU Splitting

1. INTRODUCTION

HEVC [1] was developed by the JCT-VC to replace to its predecessor, H.264/AVC standard. The main goal of HEVC was to signicantly improve the Rate and Distortion (RD) performance compared to H.264/AVC in order to allow for new applications, such as beyond High Definition (HD) resolutions (so called 4K, 3840x2160 pixels, and 8K, 7680x4320 pixels). It is surely the most significant event in digital video compression field in a decade. With the collaborative effort of a lot of experts, HEVC can provide approximately twice the compression performance of prior standards while maintaining the same level of video quality but at a cost of extremely higher computational and storage complexities [2].

Considering both, the superior compression performance of HEVC, as well as the large body of content that are currently encoded using H.264/AVC standard, a transcoder that can convert H.264/AVC bitstreams into HEVC bitstreams has a great value in many applications, especially before dedicated HEVC encoder systems become widely available, while at the same time, various software based HEVC decoders have been demonstrated [3]. Furthermore, there is a wide availability of H.264/AVC encoders in the market with a good tradeoff in terms of RD performance and low cost. Thus, a H.264/AVC encoder working in tandem with an efficient H.264/AVC to HEVC transcoder may provide a cost-effective mean of conducting HEVC encoding for many applications in the absence of dedicated HEVC encoders. Therefore, the motivation for a H.264/AVC to HEVC transcoder is twofold: 1) to be ready to promote interoperability for the legacy video encoded in H.264/AVC format when new applications using the HEVC emerge and 2) to be able to take advantage of the superior RD performance of the HEVC.

With this challenge in mind, this paper presents an algorithm to be used as part of a low complexity heterogeneous H.264/HEVC video transcoder that greatly reduces the complexity of the transcoding process. The proposal tries to exploit the information gathered in the H.264/AVC decoder to assist decisions on CU splitting in HEVC using a statistical NB classifier to avoid exhaustive RD Optimization (RDO) search on all possible CU sizes and its modes. Experimental results show that the proposed algorithm can achieve a speed-up in the video transcoder up to 3.98 without significant loss in the RD performance and it also improves some of the related works available in the literature.

The remainder of this paper is organized as follows: Section 2 includes a technical background and the related work which is being developed about the topic. Section 3 introduces our proposed transcoding algorithm. Experimental results are shown in Section 4. Section 5 concludes the paper and includes the future work.

This work has been jointly supported by the MINECO and European Commission (FEDER funds) under the projects TIN2012-38341-C04-04, TIN2010-20900-C04-03 and TIN2013-46638-C3-3-P. Likewise, this work has also been supported by the Spanish Education, Culture and Sports Minister under grant FPU12/00994.

2. RELATED WORK

HEVC introduces new coding tools respect to its predecessor H.264/AVC as well as it improves other which were already used [1][2]; all of them make it possible to notably increase the coding efficiency. One of the most important changes affects to the picture partitioning. HEVC dispenses with the terms of Macro-Block (MB) and Block for the Motion Estimation (ME) and the transform respectively and introduces three new concepts: CU, Prediction Unit (PU) and Transform Unit (TU). This structure leads to a more flexible coding to suit the particularities of the frame. Each picture is partitioned into squared regions of variable size called CUs which replace the MB structure of previous standards. Each CU, whose size is limited from 8x8 to 64x64 pixels, may contain one or several PUs and TUs. To fix the size of each CU, first of all, a picture is divided into 64x64 pixels areas, which are called Coding Tree Units (CTU), and then, each CTU can be partitioned into 4 smaller sub-areas of a quarter of the original area. This partitioning can be done with each sub-area recursively until it has a size of 8x8 pixels.

HEVC checks most of PUs (Inter and Intra modes) to decide whether it splits a CU or not by choosing the best RD case. Furthermore, in the case of inter prediction, for each of these PU partitions a ME algorithm is called. This wide range of possibilities makes HEVC to be much more computationally expensive than its predecessor H.264/AVC. HEVC introduces changes in other modules too, such as the Intra Prediction (where a whole of 35 different coding modes can be selected), the PU modes (it introduces asymmetric modes), new image filters or new transform sizes [1][2].

On the other side, video transcoding is the process of converting a compressed video stream encoded with a determinate format or characteristics into another video stream encoded with a different codec or characteristics. The transcoding process should perform the conversion without making necessary the complete process of decoding and re-encoding [4]. Transcoding has been a hot research topic in the last years in the framework of MPEG-2 to H.264/AVC [5] or H.263 to H.264/AVC transcoders [6], also between H.264-extensions such as H.264/AVC to SVC [7] or, even, between Distributed Video Coding (DVC) and H.264/AVC [8].

As far as the authors of this paper know, nowadays, there are few approaches that deal with the problem of converting already encoded streams with H.264/AVC into the new standard HEVC. The approach presented in [9] focuses on reducing the number of CUs and PUs partitions to be checked by means of an improved RDO metric. In [10], the authors propose a reuse of Motion Vectors (MVs) as well as a similarity metric to decide which HEVC CU partitions have to be tested. In [11] proposal which combines parallelization and reutilization of information fetched at the decoder side of H.264/AVC can be found. This second part of the proposal uses the frame resolution in order to restrict the quadtree splitting and it reuses the partitioning modes in order to select the available PU modes in each case. In [12] the k first frames of the sequence are used to compute the parameters so that the transcoder can learn the mapping for that particular sequence, is proposed. Then, two types of mode mapping algorithms are proposed. In the first solution, a single H.264/AVC coding parameter is used to determine the outgoing HEVC partitions using dynamic thresholding. The second solution uses linear discriminant functions to map the incoming H.264/AVC coding parameters to the outgoing HEVC partitions; this solution is called Proposed Transcoder for Content Modeling using Linear Discriminant Functions (PTCM-LDF).

In this paper, a novel technique for accelerating the HEVC partitions is proposed based on a Bayesian Classifier; in particular the simplest one is used, NB classifier (see e.g. [13]). The existence of a correlation between some information from H.264/AVC (residual, motion vectors, modes...) and HEVC partitions is considered to learn a classifier to be used for the selection of the best CU partition. This technique converts a very complex process into a simple set multiplications of probabilities, which significantly reduces the complexity of the HEVC encoder, as shown by the results presented.

3. PROPOSED TRANSCODER

This paper proposes an algorithm which aims to reduce the computational complexity to decide the appropriate depth for each quadtree. For each level, the algorithm decides whether it is more likely to split the CU (C_S) or not (C_N). Therefore, C_S and C_N are the labels of the class variable to predict by the proposed approach. Figure 1 describes the CU splitting algorithm for the proposed transcoder, which we have called FQLD. The idea resides on determining whether the CU has to be partitioned or not. If C_S is chosen, only Skip and 2Nx2N PUs are checked and, otherwise, all PUs at this CU depth are evaluated and the algorithm for this CTU finishes.

In this study we focus on the use of machine learning (supervised classification) at levels 0 and 1, while at level 2 a much simpler approach is followed. Basically, as CU size at this level is 16x16 pixels, which is the MB size in H.264/AVC, the proposed algorithm simply mimics H.264/AVC: if MB size was 16x16, 16x8 or 8x16, then the decision is C_N, otherwise (smaller sizes), the decision is C_S.

At levels 0 and 1 of the quadtree (CU sizes of 64x64 and 32x32 pixels respectively) probabilistic classifiers are learnt and used to make the decision, in particular, a NB classifier has been selected since this model is computationally efficient while achieving a high hit rate. A NB classifier computes the posterior probability of each label C_i given the set of features $\mathbf{F} = \{w_1, \ldots, w_n\}$ as input: $P(C_i|\mathbf{F}) \propto P(C_i) \prod_{j=1}^{n} P(w_j|C_j)$, and it decides the output label as one with high probability.

Four NB classifiers for levels 0 and 1 and frame types P and B (M_{0P}, M_{1P}, M_{0B} and M_{1B}) have been trained offline

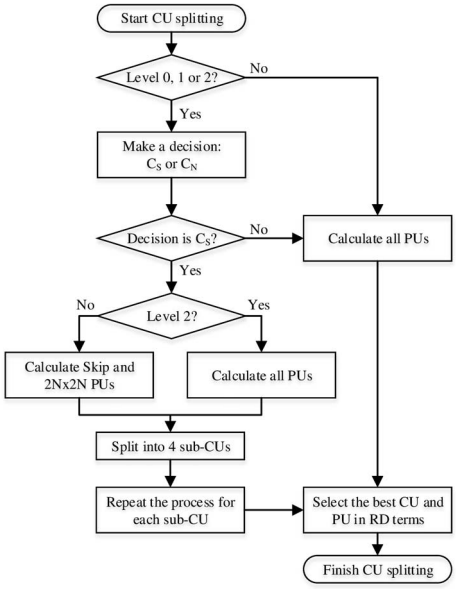

Fig. 1. Diagram of the proposed FQLD algorithm.

using $QP = \{22, 27, 32, 37\}$ and 6 sequences from those described in [14] (*PeopleOnStreet, ParkScene, BQMall, RaceHorses, Johnny* and *SlideEditing*). The first 1000 CUs for each QP-sequence pair were selected. The initial set of features, **F**, fetched from H.264/AVC stream, are calculated for the covered area (in MBs) by the current HEVC CU:

- w_{QP}: QP value used to encode the stream.
- w_{bits}: amount of bits used to encode all the MBs for the current CU after applying the Context-adaptive binary arithmetic coding (CABAC) operation.
- w_{intra}, w_{skip}, w_{16}, w_4, w_{inter}: amount of Intra, Skip, Inter 16x16, Inter 4x4 and other Inter MBs respectively.
- w_{DCTno0}: number of non-zero DCT coefficients.
- w_{width}, w_{height}: frame width and height respectively.
- w_{MVsum}: sum of all the MVs components contained in the frame.
- w_{resAvg}, w_{resVar}: average and variance of the residue for the covered area respectively.
- $w_{resAvgSubCU1}$, $w_{resAvgSubCU2}$, $w_{resAvgSubCU3}$, $w_{resAvgSubCU4}$: average of the residue for each sub-CU: 1, 2, 3 and 4 respectively.
- $w_{resVarSubCUs}$: variance of the 4 previous values.
- w_{sobelH}, w_{sobelV}: sum of applying Sobel operator [15] in the residue in horizontal and vertical directions.
- w_{MVxAvg}, w_{MVyAvg}, w_{MVxVar}, w_{MVyVar}: average and variance of x and y MVs components respectively for the covered area.

Starting from this original set of features, four independent machine learning processes are carried out in order to learn the corresponding models, M_{0P}, M_{1P}, M_{0B} or M_{1B}. The steps of the learning process are:

1. To avoid the standard (and improbable) assumption that the values of each feature given each class label follows a parametric distribution, we discretize all the numerical variables by using a supervised method [16], that is, the intervals are chosen in such a way, that the resulting *discrete* variable has as much as possible discriminative power regarding the class variable.

2. After that, a wrapper stepwise forward attribute selection process [17] was carried out to select the *best* subset of features according to the prediction of the class labels. NB algorithm is used during the search to evaluate the goodness of each subset, removing those redundant and irrelevant variables that may hurt its accuracy.

3. From the resulting discretized and reduced dataset a NB classifier is learnt, and posteriorly calibrated in order to accommodate the particularities of the addressed problem. That is, since 64x64 and 32x32 CUs are not usually chosen, a cost analysis has been carried out by increasing the cost of choosing C_N when it really was C_S. This cost has been set based on the quadtree level and the resolution (higher resolutions tend to have higher CU sizes and viceversa): on the one hand, for level 0 the cost is always 2.0 and for level 1 the cost is also 2.0 if the resolution is smaller than Full HD and 1.0 in other case. On the other hand, the cost of choosing C_S when it really was C_N is always 1.0 and the cost of correct classification is 0.0.[1]. The output from this analysis is the threshold to be used in the classification, instead of the standard 0.5 one.

4. PERFORMANCE EVALUATION

The transcoder has been tested with the sequences and test conditions approved in [14]. Used QP values were $\{22, 27, 32, 37\}$ and configurations were Random Access Main 10 (RA), Low Delay B Main 10 (LB) and Low Delay P Main 10 (LP). The results shown are the average values of the sequences which compose a class. The software used has been JM 18.4 [18] for H.264/AVC and HM 12.0 [19] for HEVC. The remainder of coding parameters are kept as default in the configuration file. Thus, the process to generate this results is the following:

1. Encode the YUV file with H.264/AVC reference software using *HM-like* configuration files.
2. Decode each file with the decoder side of the proposed transcoder, producing a YUV' file as well as all the information needed for the proposed algorithm.

[1]A value of 2.0 means that we have estimated that the cost of wrongly classifying C_S as C_N is twice that the contrary error. This type of error at high levels (0 or 1) can have a great impact in the output sequence, because subsequent low-level splitting is not considered.

Table 1. Transcoder with FQLD algorithm results

Configuration	Sequence class	Level 0		Levels 0 and 1		Levels 0, 1 and 2	
		BD-rate (%)	Speed-up	BD-rate (%)	Speed-up	BD-rate (%)	Speed-up
RA	Class A	0.3	1.24	7.5	2.19	8.0	2.60
	Class B	0.2	1.29	6.4	2.41	5.5	2.23
	Class C	0.2	1.19	2.5	1.75	4.0	2.11
	Class D	0.4	1.19	1.9	1.79	2.9	2.10
	Average	*0.3*	*1.23*	*4.7*	*2.02*	*5.1*	*2.24*
LB	Class B	0.2	1.12	5.9	2.25	6.1	2.17
	Class C	0.0	1.11	2.7	1.66	3.8	2.09
	Class D	0.0	1.11	1.8	1.72	3.7	2.16
	Class E	1.1	2.04	5.6	3.61	6.0	3.98
	Average	*0.2*	*1.22*	*4.0*	*2.05*	*4.9*	*2.35*
LP	Class B	0.1	1.11	4.6	1.79	5.4	2.29
	Class C	0.1	1.06	2.2	1.56	3.7	2.01
	Class D	0.1	1.17	1.3	1.46	2.8	1.88
	Class E	1.0	1.65	5.4	3.24	6.4	3.86
	Average	*0.3*	*1.18*	*3.3*	*1.77*	*4.5*	*2.26*
Global average		*0.3*	*1.21*	*4.0*	*1.95*	*4.8*	*2.28*

3. Encode the YUV' file with the encoder side of the original transcoder (anchor).
4. Encode the YUV' file with the encoder side of the proposed transcoder (proposed).
5. Compare the anchor and the proposed streams in order to obtain the BD-rate and the speed-up.

Table 1 contents the results for RA, LB and LP configurations in terms of speed-up and BD-rate [20] (which measures the increment in bitrate while maintaining the same objective quality). This table shows the difference of applying the FQLD algorithm to incremental levels so that the evolution of the speed-up and BD-rate can be appreciated. Thus, it can be seen that the more levels the algorithm is applied, the greater are the speed-up and the BD-rate. Moreover, results show that level 0 could achieve a moderate speed-up without a significant loose. Therefore, the quality-complexity could be adjusted by the user deciding whether to apply the FQLD algorithm to 1, 2 or 3 levels.

Regarding configurations, it can be observed that RA is, in average, the configuration with the greatest time saving, since it is the one with most complexity in the ME module. LB configuration obtains a similar speed-up since the algorithm works well with E class as it contains videoconferencing sequences, which are easy to predict due to static backgrounds and few details. Finally, LP configuration obtains less speed-up since the ME module has less weight in P frames.

4.1. Comparison with other proposals

Table 2 shows the comparison between FQLD algorithm and PTCM-LDF algorithm (proposed by [12] and depicted in Section 2), which can be parametrized with the number of frames used to train and the length of the sequence. C class and the same configuration than in [12] is used (LP configuration with only 1 reference frame). It can be seen that for comparable speed-ups, FQLD achieves lower BD-rates.

Table 2. Comparison between PTCM-LDF and FQLD algorithms

			BD-rate (%)	Speed-up
PTCM-LDF	10 frames	2.5 s	4.89	1.95
		5 s	4.88	2.04
		10 s	5.42	2.15
	25 frames	2.5 s	4.45	1.69
		5 s	4.86	1.93
		10 s	5.82	2.10
	50 frames	2.5 s	3.34	1.44
		5 s	4.57	1.85
		10 s	6.03	1.95
	Average		4.92	1.90
FQLD			**4.00**	**1.89**

5. CONCLUSIONS AND FUTURE WORK

This paper contains a proposal of algorithm which can accelerate the transcoding process from H.264/AVC to HEVC by deciding which quadtree level is the most appropriate without the need of testing all the possible CUs/PUs. Results show that a good tradeoff between quality loss and acceleration is achieved: a 2.28 of speed-up in average with a negligible increment of a 4.8% in the BD-rate. Moreover, FQLD can achieve better BD-rates than a state of the art algorithm such as PTCM-LDF when similar speed-ups are compared.

As future work, dynamic solutions which would adjust the bayesian model to a particular sequence could be explored. Moreover, the set of features could be improved using information from the Skip and 2Nx2N PUs.

6. REFERENCES

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

[2] 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)," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 22, no. 12, pp. 1669–1684, Dec 2012.

[3] C. C. Chi, M. Alvarez-Mesa, B. Juurlink, G. Clare, F. Henry, S. Pateux, and T. Schierl, "Parallel scalability and efficiency of hevc parallelization approaches," *Circuits and Systems for Video Technology, IEEE Transactions on*, vol. 22, no. 12, pp. 1827–1838, Dec 2012.

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

[5] G. Fernandez-Escribano, H. Kalva, P. Cuenca, L. Orozco-Barbosa, and A. Garrido, "A fast mb mode decision algorithm for MPEG-2 to H.264 p-frame transcoding," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 18, no. 2, pp. 172–185, Feb 2008.

[6] G. Fernandez-Escribano, J. Bialkowski, J.A. Gamez, H. Kalva, P. Cuenca, L. Orozco-Barbosa, and A. Kaup, "Low-complexity heterogeneous video transcoding using data mining," *IEEE Transactions on Multimedia*, vol. 10, no. 2, pp. 286–299, Feb 2008.

[7] R. Garrido-Cantos, J. De Cock, J.L. Martinez, S. Van Leuven, and P. Cuenca, "Motion-based temporal transcoding from H.264/AVC-to-SVC in baseline profile," *Consumer Electronics, IEEE Transactions on*, vol. 57, no. 1, pp. 239–246, February 2011.

[8] A. Corrales-Garca, J. L. Martinez, G. Fernndez-Escribano, and F. J. Quiles, "Variable and constant bitrate in a DVC to H.264/avc transcoder," *Signal Processing: Image Communication*, vol. 26, no. 6, pp. 310–323, 2011.

[9] D. Zhang, B. Li, J. Xu, and H. Li, "Fast transcoding from H.264 avc to High Efficiency Video Coding," in *IEEE International Conference on Multimedia and Expo (ICME) 2012*, July 2012, pp. 651–656.

[10] E. Peixoto and E. Izquierdo, "A complexity-scalable transcoder form H.264/AVC to the new HEVC codec," in *International Conference on Image Processing (ICIP), Orlando, FL, USA (September 2012)*, 2012.

[11] 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*, March 2013, pp. 241–250.

[12] E. Peixoto, T. Shanableh, and E. Izquierdo, "H.264/AVC to HEVC video transcoder based on dynamic thresholding and content modeling," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 24, no. 1, pp. 99–112, Jan 2014.

[13] J. Flores, J. A. Gámez, and A. M. Martínez, "Supervised Classification with Bayesian Networks," in *Intelligent Data Analysis for Real-Life Applications: Theory and Practice*, 2012, pp. 72–102.

[14] F. Bossen, "Common HM test conditions and software reference configurations," in *Proc. 9th JCT-VC Meeting, Geneva, Switzerland, No. JCTVC-I1100*, April 2012.

[15] S. Patnaik and Y.-M. Yang, *Soft Computing Techniques in Vision Science*, vol. 395, Springer, 2012.

[16] U. M. Fayyad and K. B. Irani, "Multi-Interval Discretization of Continuous-Valued Attributes for Classification Learning," in *Proceedings of the International Joint Conference on Uncertainty in AI*, 1993, pp. 1022–1027.

[17] I. Guyon and A. Elisseeff, "An introduction to variable and feature selection," *Journal Machine Learning Research*, vol. 3, pp. 1157–1182, 2003.

[18] Joint Collaborative Team on Video Coding, *Reference Software to Committee Draft, version 18.4*, 2012.

[19] K. McCann, B. Bross, W.-J. Han, I. K. Kim, K. Sugimoto, and G. J. Sullivan, "High Efficiency Video Coding (HEVC) test model 12 (HM 12) encoder description," in *Proc. 14th JCT-VC Meeting, Vienna, Austria, No. JCTVC-N1002*, July 2013.

[20] G. Bjontegaard, "Improvements of the BD-PSNR model," *ITU-T SG16 Q*, vol. 6, pp. 35, July 2008.