Fast CU Partition Strategy for HEVC Intra-Frame Coding Using Learning Approach via Random Forests

Bochuan Du, Wan-Chi Siu and Xuefei Yang
Centre of multimedia signal processing, Department of Electronic and Information Engineering
The Hong Kong Polytechnic University, Hong Kong
E-mail: enwcsiu@polyu.edu.hk Tel: +852-2766 6229

***Abstract*—HEVC (High Efficiency Video Coding) achieves cutting edge encoding efficiency and outperforms previous standards, such as the H.264/AVC. One of the key contributions to the improvement is the intra-frame coding that employs abundant coding unit (CU) sizes. However finding the optimal CU size is computationally expensive. To alleviate the intra encoding complexity and facilitate the real-time implementation, we use a machine learning technique: the random forests, for training. Based on off-line training, we propose using the forest classifier to skip or terminate the current CU depth level. In addition, neighboring CU size decisions are utilized to determine the current depth range. Experimental results show that our proposed algorithm can achieve 48.31% time reduction, with 0.80% increase in the Bjøntegaard delta bitrate (BD-rate), which are state-of-the-art results compared with all algorithms in the literature.**

I. INTRODUCTION

In 2013, the Joint Collaborative Team on Video Coding (JCT-VC) has come up with a milestone of video compression standard, known as High Efficiency Video Coding (HEVC) [1], which is an evolved version of H.264 [2]. HEVC outperforms H.264 by increasing the compression rate by about 50% and obtaining a high encoding efficiency for resolution beyond High Definition (HD) videos, such as the latest 4K and 8K formats videos[3][4]. Compared with H.264, HEVC has two main enhancements in encoder, i) nested quad-tree structures with coding unit (CU), prediction unit (PU), and transform unit (TU) [5] and ii) numerous prediction modes [4].

One of the considerable areas in HEVC is intra coding, for which blocks of pixels are coded only with references to the current frame. It exploits the spatial correlation on blocks with 35 prediction modes and CU sizes varying from block size 64x64 down to 8x8 [6]. These new features affect the complexity extremely, making it difficult for real-time implementation, especially for HD or Ultra HD videos that have high resolutions.

The fact motivates researches on early decision of CU size to reduce the coding complexity and achieve a fast encoding speed. Authors in [7] proposed an online updated statistical parameter to set the threshold for early decision on CU size splitting and pruning. In [8], texture measurements on each CU block are utilized to bypass and terminate CU size decision. Edge complexity in several directions are used to decide the partitioning of a CU in [10]. However, most of these existing algorithms mainly apply conventional content-based or statistic-based methods. Limited number of papers in the literature consider to used learning-based algorithms, which is currently a hotspot and widely applied in recognition, classification and signal processing. One of the latest literature that uses learning-based method on CU size decision in intra frame prediction is [9], where data mining classifiers are used. But the time saving ratio might be far from the state-of-the-art results and the testing sequences used in the paper appear incomplete for a full evaluation. Many papers in literature that explore fast intra coding algorithms via prediction mode selection [11][12], which is also a hot topic for intra prediction research, while solely applying fast prediction mode selection usually cannot give an effective complexity reduction in HEVC.

In this paper, we propose an efficient machine learning algorithm based on fast CU depth decision, where we mainly apply random forests for the learning. For the rest of this paper, we will give a brief introduction of HEVC intra-frame prediction in section II, mainly focusing on CU size decision. Section III covers the details of our proposed algorithm, which includes CU range decision and the fast split or non-split strategy. In section IV, we will show and compare experimental results. Finally, the conclusion and references are provided.

II. INTRA PREDICTION IN HEVC

Intra prediction is a block-based prediction. Compared with the H.264 standard where three block sizes with four (for 16x16 block size) and nine (for 8x8 and 4x4 block size) prediction modes are available to predict a block within the current frame [1], the HEVC not only enlarges the block sizes to 64x64, 32x32, 16x16, 8x8 and 4x4, but also increases the set of prediction modes to 35 [2].

HEVC starts the prediction from a Coding Tree Unit (CTU) whose size is 64x64. It will then be further split into four 32x32 CUs, each of the four sub-CUs can also be split iteratively into a smaller quad-tree-based structure [6]. Figure 1 gives an example of the CTU size decision on both the tree structure and corresponding block partitions. Depth 0 refers to the block size 64x64. And the number 0 and 1 represent the

non-split and split decision. If the cost of prediction on the four 32x32 blocks is lower than the cost of predicting 64x64 block, the depth level will be increased by one and goes into the analysis of each 32x32 blocks. Similarly, a 32x32 CU will be compared with its four 16x16 sub-CUs. When it comes to 8x8 CU, it may be divided into PUs with size of 4x4. Finally, the partitioning ends until the minimum cost in each level is found.

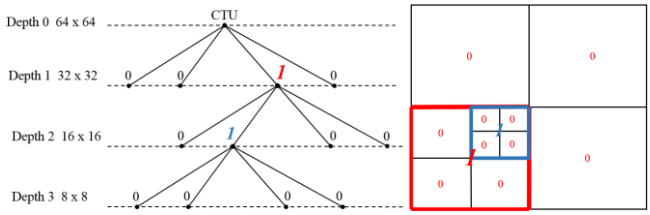

Fig. 1 An example of CU partitions

In each CU, 35 prediction modes, including 2 modes for smooth texture prediction and 33 angular prediction modes, are compared by Rough Mode Decision (RMD) [13][14][15] and Rate Distortion Optimization (RDO) [6]. Then the optimum mode with the lowest cost is obtained.

Due to numerous amount of possibilities to be evaluated, HEVC is time-consuming. This fact evokes our approach to realize an early CU size decision without going through the tree structure.

III. Our Proposed Fast CU Decision

A. CU Depth Range Decision

In a natural scene, neighboring blocks have a strong correlation with the current CU. It is known that the intra-frame coding makes use of the encoded above, left, left-above and right-above CUs as shown in Figure 2 to reduce computational burden in the current CU.

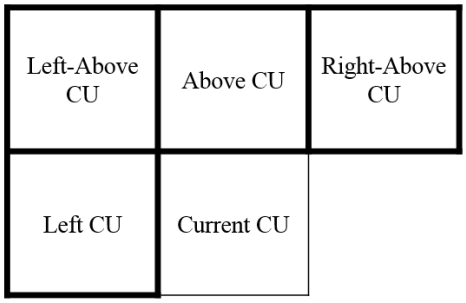

Fig.2 Current CU and its Neighbor Depth Decision

These neighboring CUs usually have similar contents, but HEVC yet has no good methods to aid the current CU size decision by utilizing its neighbors. We can make use of the spatial correlation of them, and predict a depth level as below.

$$Depth_{pred} = \sum_{i=0}^{M-1} w_i \cdot Depth_i \qquad (1)$$

where $Depth_i$ is the depth level of the neighboring CUs, ranging from 0 to 3. M is the number of neighboring blocks, and we have $M=4$, including the left CU, left-above CU, above CU and right-above CU. w_i is the weighting factor determined by the correlation between the current CU and each neighboring CU. The sum of these four weights is normalized to 1. Due to a stronger correlation, we assign $w_i = 0.3$ for the left CU and above CU. For the left-above CU and right-above CU which have weaker correlation with the current CU, we let $w_i = 0.2$. This arrangement is for the sake of simplicity.

After obtaining the $Depth_{pred}$ the current CU depth range is assigned as below.

$$\text{Depth Range} = \begin{cases} 0-2, if\ 0 < Depth_{pred} < 1 \\ 0-3, if\ 1 \le Depth_{pred} \le 2 \\ 1-3, if\quad Depth_{pred} > 2 \end{cases} \qquad (2)$$

When the predicted depth is smaller than 1, it means the neighboring blocks are flat and homogeneous. The current depth range can be set from 0 to 2, where the smallest CU size for prediction is limited to 16x16 (depth 2). When the predicted level is larger than 2, this reveals that the neighboring CUs are complex and inhomogeneous. We will set the prediction range from 1 to 3, which means the current CU prediction starts from block size 32x32 (depth 1), and searches towards 8x8 (depth 3). For the rest cases of the predicted depth, the prediction range remains unchanged, which is the same as the default HEVC settings.

This approach is similar to reference [8] and [16], except that [8] is limited to bypass the current block size when the current depth level is smaller than the predicted depth minus one. And [16] defines the depth range by considering the ranges of the left and above CU only.

B. Early CU Termination and Bypass Decision by Random Forests

In this section, we introduce the core of our proposed algorithm. First, our forest design is given with a brief introduction of decision tree and random forests. Then a split algorithm is introduced for the optimum split decision. Lastly, we will provide detailed strategies for the training, and the evaluations of the training results.

1) Forest Design

The approach random forests [17]-[20], $T=\{T_t\}$, is a combination of a set of decision trees $\{T_t\}$ which jointly realize classification. Each tree has three types of nodes:

- A **root node** that has no incoming edges with zero or more outgoing edges.
- **Internal nodes**, each of which has exactly one incoming edge and two or more outgoing edges.
- **Leaf** or **Terminal nodes**, each of which has exactly one incoming edge and no outgoing edges.

The prediction starts with the root node. For each **non-leaf node**, which includes the root and internal nodes, the procedure allows an incoming image block to go to the left or to the right based on the value of a certain variable whose index is stored in the current node. The tree is built recursively, at each node the recursive procedure may stop when:

- Depth of the tree branch has reached the maximum defined value.
- Number of training samples is less than a specified threshold when further split is not statistically representative.
- All the samples belong to the same class.

In our designed forests, all the trees are trained with different training sets but with the same set of parameters. These sets are generated from the original training set using the bootstrap procedure. We totally uses 10 trees, where each of them has a maximum depths of 25. We apply the luma pixel intensity in each block as the input feature vector, and the output classification is "0" for "non-split" decision and "1" for "split" decision. As shown in Figure 3, the classification works as below: each random tree classifier takes the input image block, classify it by going through the decision tree, and outputs the class label "1" or "0" for each tree. The final decision is made at the classification result that received the majority of "votes" from all the 10 outputs.

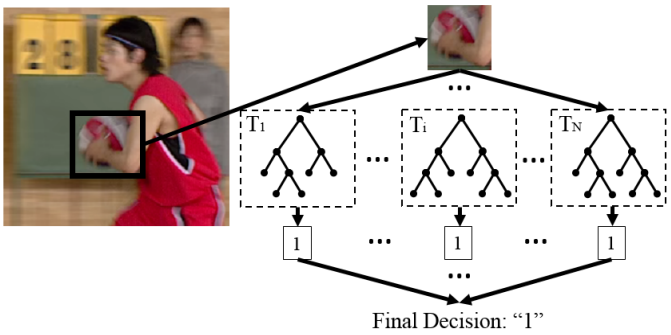

Fig. 3 Random forest with a set of decision trees.

2) Split Algorithm

In the training of each non-leaf node, a set of thresholds is tested. If the pixel value is less than the threshold, the image block goes to the left, otherwise it goes to the right. The optimum threshold can be obtained by many measures. In our experiments, we use the Gini impurity criterion [19]

$$Gini(t) = 1 - \sum_{i=0}^{c-1} [p(i \mid t)]^2 \qquad (3)$$

where c is the number of classes (here we have $c=2$), $p(i|t)$ denotes the ratio of data belonging to class i at the node t. If the Gini impurity is lower, the classification performance is better. Then, by going through all possible thresholds, the optimum one is obtained with the lowest Gini impurity. For example, there are two thresholds Th_1, Th_2 need to be tested. Threshold Th_1 results in 10 training samples going to the left node, where 8 of them are classified as "0", and the rest 2 of them are classified as "1". The Gini impurity for Th_1 is

$$Gini_{Th1} = 1 - (\frac{8}{10})^2 - (\frac{2}{10})^2 = 0.32 \qquad (4)$$

Threshold Th_2 results in 20 training samples going to the left node, where 10 of them are classified as "0", the other 10 samples are classified as "1". The Gini impurity for Th_2 is

$$Gini_{Th2} = 1 - (\frac{10}{20})^2 - (\frac{10}{20})^2 = 0.50 \qquad (5)$$

Since $Gini_{Th1} < Gini_{Th2}$, Threshold Th_1 is then selected as the optimum threshold for the split.

3) Training Scheme and Evaluation

Feature selection aims to select attributes that can realize a high classification performance. In our proposed method, we take the luma pixel intensity as the input attributes for the CU size 64x64 and 32x32 determination. The classification result is set to "0" which means to stay in the current depth without any further split and "1" as to skip the current CU depth and start the prediction from one depth higher. As for a block belongs to "0", it has a homogenous texture or a smooth grain that follows a certain direction. In this way, the block can be easily predicted as one of the 35 prediction modes and results in tiny residual and RD cost. On the other hand, for the block belongs to "1", it owns a varied texture and further split is required. The detailed parameters for our training is as below.

TABLE I
TRAINING PARAMETER SETTINGS FOR BLOCK DEPTH 0 AND 1

Parameters	Numbers
Number of training samples	2400 for block depth 0 7600 for block depth 1
Attributes per sample (block size)	4096 (=64x64) for block depth 0
Number of Classification	2 ("0 for non-split", "1" for split)
Max depth	25
Minimum number of training sample for further split	5
Number of variable randomly selected at node and used to find the best splits	64 for block depth 0 32 for block depth 1
Number of trees in the forest	10

The number of training samples is set to the extent of a high prediction accuracy without overfitting. The number of attributes per sample, which means the number of input feature vectors per sample, is 4096 for image block size 64x64 and 1024 for image block size 32x32. Then the classification decision is assigned as the last number in a sample, where "0" is for non-split decision and "1" for split decision. The maximum depth of a decision tree is set to 25, which is obtained by experiments from setting the maximum depth as 5, 10, 15, 20... The minimum number of training samples for further split is set to 5. And the number of variables randomly selected at node and used to find the best split is 64 for block size 64x64, and 32 for block size 32x32, which is useful to explore some texture characteristics. Finally, we set 10 as the number of trees in the forest.

Considering the optimal partitioning obtained based on pixel variation in the intra-frame prediction has strong dependency with the scene and the QP value, we trained the classifier sequence by sequence with four QP values: 22, 27, 32, and 37 as defined in JCT-VC [21]. The ground truth classification is obtained by encoding the HEVC testing model HM10.0 [22]. Table II and Table III show part of our results about the prediction accuracy on the first 200 frames is of Class B sequences. From the tables above, we can find that

this method achieves a high prediction accuracy, where most of the results can reach an accuracy of over 90%. The results can guarantee a fast and efficient depth decisions. For the blocks with wrong classification, there might be some degradation in PSNR and increase in BD-rate. But even the size of CU is wrongly predicted, for example, the ground truth is to stay at depth 0. Our decision stops at depth 2 as shown in Figure 4. The quality may not be affected severely because further split can also achieve an accurate prediction. With the low wrong prediction rate, the degradation of picture quality should be low.

TABLE II
PREDICTION ACCURACY OF CU DEPTH 0 OF CLASS B

CU Depth 0 Prediction Accuracy						
QP	Average Training Time (s)	Testing Sequence				
		Basketball	BQTerrace	Cactus	Kimono	Park
		Accuracy (%)	Accuracy (%)	Accuracy (%)	Accuracy (%)	Accuracy (%)
22	13.46	99.40	96.80	99.60	99.80	98.10
27	14.30	94.70	97.40	98.70	95.30	97.60
32	15.79	90.10	98.30	97.20	97.00	98.30
37	18.66	89.30	99.40	98.30	98.30	97.40

TABLE III
PREDICTION ACCURACY OF CU DEPTH 1 OF CLASS B

CU Depth 0 Prediction Accuracy						
QP	Average Training Time (s)	Testing Sequence				
		Basketball	BQTerrace	Cactus	Kimono	Park
		Accuracy (%)	Accuracy (%)	Accuracy (%)	Accuracy (%)	Accuracy (%)
22	15.22	90.90	97.20	93.80	97.10	90.80
27	18.01	90.60	96.40	95.20	97.20	93.80
32	20.08	91.10	95.20	96.20	98.10	93.50
37	21.19	86.60	89.70	97.50	97.50	94.80

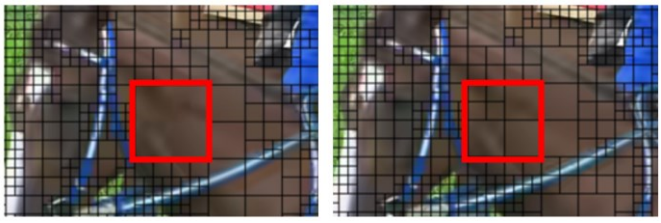

Fig.4 Comparison on CU Size Decision.
(Left: ground truth CU size encoded by HM10.0. Right: predictions of our algorithm, all matching well, except the CU encoded in red.)

C. *Flowchart of our Proposed Algorithm*

Figure 5 sums up our proposed algorithms as illustrated in section III.A and III.B. When CU encoding starts, the depth range decision will firstly be found by making use of neighboring blocks. Then, if the depth level of the current CU is 0 or 1, we use random forests classifiers to decide whether the prediction should be bypassed or stayed on the current depth level. For the rest of CU depths (level 2 and 3), conventional HEVC intra prediction methods will be used.

IV. EXPERIMENTAL RESULT

The proposed fast algorithm is implemented on the HEVC reference software HM10.0 reference software [22], and was run on a platform with Intel® Core™ i7-4790 CPU 16.0GB RAM size. According to the common test conditions and reference configurations recommended by JCT-VC [21], five groups of testing sequences were carried out to evaluate the coding performance and computational complexity under the

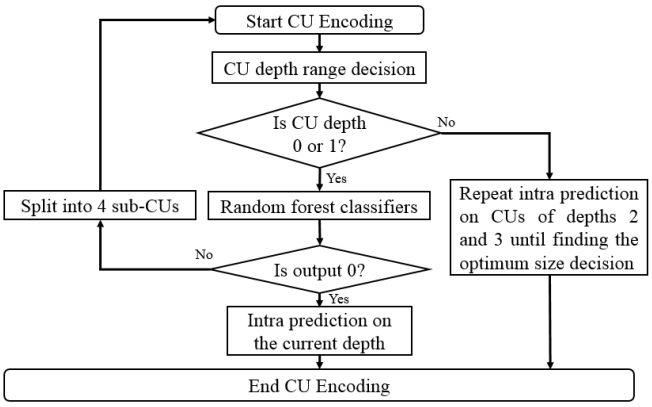

Fig.5 Flowchat of the proposed algorithm

configuration of All-Intra Mode and Main profile (AI-Main). Four QP values (QP 22, QP 27, QP 32, and QP 37) were used to encode all the frames in the test sequences. For each QP of a sequence, we off-line trained two forests in advance, which follows our methodology introduced in section 3.2.2. The Bjøntegaard delta rate distortion (BD-rate) performance is measured by BD-rate Model [23] [24]. The average time ratio (ATR) and average time saving (ATS) are given by:

$$ATR(\%) = \frac{1}{n}\sum_{i=1}^{n}\frac{Enc.Time_{proposedi}}{Enc.Time_{HM10.0,i}} \quad (6)$$

$$ATS(\%) = \frac{1}{n}\sum_{i=1}^{n}\frac{Enc.Time_{HM10.0,i} - Enc.Time_{proposedi}}{Enc.Time_{HM10.0,i}} \quad (7)$$

where n is the total number of sequences, $Enc.Time_{proposed,i}$ is the encoding time of our proposed algorithm on sequence i, and $Enc.Time_{HM10.0,i}$ is the corresponding encoding time of HM10.0.

Table IV summarizes the performance of the overall proposed algorithm. As shown in the tables, the proposed algorithm efficiently reduces the computational complexity in terms of encoding time, with slight increase in BD-rate.

Table V gives the comparisons of the proposed algorithm with some representative methods in the literature. For a clear view of the comparison in graph, we use ATR to reveal the complexity reduction on encoding time. As shown in Figure 6, the smaller value in X axis, which means the lower ATR, the more time is saved to encode. And the smaller value in Y axis, which reveals the lower BD-rate, the less additional bits are used to encode. Therefore, the nearer to the left-down corner in the graph, the better is the performance of the algorithm. Our proposed algorithm has a good performance and is comparable to [7] which is the state-of-the-art result. Furthermore, our proposed algorithm outperforms [8] in both BD-rate and time ratio, where [8] also has a good result. In addition, [9] is one of very few algorithms that uses machine learning technique. Although its BD-rate is a little bit lower than our result, our complexity gains a 20% reduction. Compared with [25] and [26], our result also has advantages in both BD-rate and time saving.

TABLE IV
OVERALL RESULTS OF THE PROPOSED ALGORITHM

Classification	Sequence	Num. of frames	All Intra Main	
			BD-rate (%)	ATS (%)
Class A (2560x1600)	Traffic	150	0.9	44.8
	PeopleOnStreet	150	0.6	40.6
	Nebuta	300	1.1	63.4
	SteamLocomotive	300	1.7	78.1
Class B (1920x1080)	Kimono	240	1.8	76.1
	ParkScene	240	0.6	47.0
	Cactus	500	0.7	45.8
	BasketballDrive	500	1.8	63.4
	BQTerrace	600	0.3	47.8
Class C (832x480)	BasketballDrill	500	0.6	38.1
	BQMail	600	0.2	35.3
	PartyScene	500	0.0	31.2
	RaceHorses	300	0.4	37.9
Class D (416x240)	BasketballPass	500	1.1	48.2
	BQSquare	600	0.1	39.9
	BlowingBubbles	500	0.2	38.2
	RaceHorses	300	0.1	33.1
Class E (1280x720)	FourPeople	600	0.6	40.0
	Johnny	600	1.9	57.1
	KristenAndSara	600	1.3	52.3

TABLE V
PERFORMANCE COMPARISON WITH EXISTING METHOD

Class	[7]		[8]		[9]		[25]		[26]		Proposed	
	BD-rate (%)	ATR (%)	BD-rate(%)	ATR (%)	BD-rate (%)	ATR (%)	BD-rate (%)	ATR (%)	BD-rate (%)	ATR (%)	BD-rate (%)	ATR (%)
Average Result	1.09	44.82	1.08	56.89	0.60	71.93	1.20	56.30	76.00	0.83	0.80	51.69

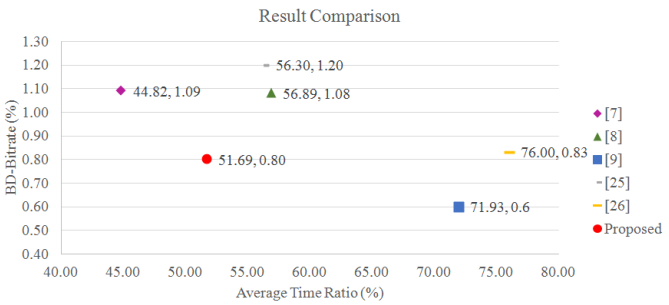

Fig. 6 Performance comparison with existing methods

V. CONCLUSION

In this paper, a fast CU depth decision of intra-frame coding based on a machine learning technique is proposed. To begin with, we have used a weighting sum approach for partition decision. This is achieved by using neighboring CUs as a reference to set the CU size range for the current CU. Then, we introduce random forests to make a prediction on selecting CU size 64x64 or 32x32. Before encoding, we input all the pixels in the image block in our random forests scheme and output the decision on whether the process stays in the current depth for prediction or directly skips the current CU size and starts the prediction for the higher depth. This is the core strategy of our algorithm which greatly reduces the computational complexity. Experimental result shows that we can achieve the state-of-the-art result, with 0.80% BD-rate increase and 48.31% time saving.

ACKNOWLEDGMENT

This work is supported by the Center for Signal Processing, the Hong Kong Polytechnic University (G-YBAS) and the Research Grant Council of the Hong Kong SAR Government: PolyU5243/13E(B-Q38S).

REFERENCES

[1] B. Bross, W.-J. Han, G. J. Sullivan, J.-R. Ohm, and T. Wiegand, "High efficiency video coding (HEVC) text specification draft 6," Joint Collaborative Team Video Coding (JCT-VC) ITU-T

VCEG and ISO/IEC MPEG, San Jose, CA, USA, April. 2012, Doc. JCTVC-H1003.

[2] T. Wiegand, G. J. Sullivan, G. Bjontegaard, and A. Luthra, "Overview of the H.264/AVC video coding standard," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 13, no. 7, pp. 560–576, Jul. 2003.

[3] J.-R. Ohm, G. J. Sullivan, H. Schwarz, T. K. Tan, T. Wiegand. "Comparison of the coding efficiency of video coding standards—including high efficiency video coding (HEVC)." *IEEE Trans. Circuits Syst. Video Technol.,* vol. 13, no. 12, pp. 1051-8215, Jan. 2013.

[4] Mahsa T., Pourazad, C. Doutre, M. Azimi, and P. Nasiopoulos. "HEVC: the new gold standard for video compression: how does HEVC compare with H. 264/AVC?" *Consumer Electronics Magazine*, vol. 1, issue 3, pp. 36-46, July, 2012.

[5] D. Marpe, H. Schwarz, and S. Bosse, "Video compression using nested quadtree structures, leaf merging, and improved techniques for motion representation and entropy coding," *IEEE Trans. Circuits Syst. Video Technol*., vol. 20, no. 12, pp. 1676–1687, Dec. 2010.

[6] J. Lainema, F. Bossen, W. J. Han, J. Min, and K. Ugur, "Intra Coding of the HEVC standard," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 22, no. 12, pp. 1792–1801, Dec. 2012.

[7] S. Cho and M. Kim, "Fast CU Splitting and Pruning for Suboptimal CU Partitioning in HEVC intra coding." *IEEE Trans. Circuits Syst. Video Technol.*, vol. 23, no. 9, pp. 1555-1694, Feb. 2013.

[8] L. Shen, Z. Zhang, and Z. Liu. "Effective CU Size Decision for HEVC Intracoding." *IEEE Trans. on Image Process.*, vol.23, no.10, pp.4232-4241, July 2014.

[9] D. Ruiz-Coll, V. Adzic, G. Fernandez-Escribano, H. Kalva, J. L. Martinez, P. Cuenca. "Fast Partitioning Algorithm for HEVC Intra Frame Coding Using Machine Learning." pp.4112-4116, Proceedings, *IEEE Intern. Confer. on Image Process. (ICIP2014)*, Paris, France, Oct. 2014

[10] B. Min and R. Cheung, "A fast CU Size Decision Algorithm for the HEVC Intra Encoder*." IEEE Trans. Circuits Syst. Video Technol.*, vol. 25, no. 5, pp. 892-896, May. 2015.

[11] L.-L. Wang, and W.-C. Siu. "Novel adaptive algorithm for intra prediction with compromised modes skipping and signaling processes in HEVC." *IEEE Trans. Circuits Syst. Video Technol*., vol.23, no.13, pp.1686-1694, Sept. 2013.

[12] L.-L. Wang, and W.-C. Siu. "H. 264 fast intra mode selection algorithm based on direction difference measure in the pixel domain." pp.1037-1040, Proceedings, *IEEE Intern. Confer. on Acoustics, Speech and Signal Process. (ICASSP)*, Taipei, Taiwan, April 2009.

[13] Liang Zhao, Li Zhang, Siwei Ma and Debin Zhao, "Fast Mode Decision Algorithm for Intra Prediction in HEVC," pp.1-4, Proceedings, *Visual Commun. and Image Process. Confer.*, Tainan, Taiwan, Nov. 2011.

[14] J.-H. Min, S. Lee, I.-K. Kim, W.-J. Han, J. Lainema, and K. Ugur, "Unification of the Directional Intra Prediction Methods in TMuC," Joint Collaborative Team Video Coding (JCT-VC) ITU-T VCEG and ISO/IEC MPEG, Geneva, Switzerland, Jul 2010. Doc. JCTVC-B100

[15] L. Zhao, L. Zhang, X. Zhao, S. Ma, D. Zhao, W. Gao. "Further Encoder Improvement of intra mode decision," Joint Collaborative Team Video Coding (JCT-VC) ITU-T VCEG and ISO/IEC MPEG, Daegu, Korea, Jan. 2011, Doc. JCTVC-D283.

[16] Y. Wang, X. Fan, L. Zhao, S. Ma, D. Zhao, W. Gao. "A Fast Intra Coding Algorithm for HEVC," pp.4117-4121, Proceedings, *IEEE Intern. Confer. on Image Process. (ICIP2014)*, Paris, France, Oct. 2014

[17] L.Breiman, "Random Forests," *Mach. Learn.*, vol. 45, no.1, pp.5-32, 2001.

[18] P. Geurts, D. Ernst, and L. Wehenkel, "Extremely randomized trees," *Mach. Learn.*, vol. 63, no.1, pp.3-42, 2006.

[19] L. Breiman, J. Friedman, C. J. Stone, and R.A. Olshen, *Classification and Regression Trees*. Boca Raton, FL, USA: CRC Press, 1984.

[20] A.Criminisi, J.Shotton, and E. Konukoglu, "Decision Forests: A unified framework for classification, regression, density estimation, manifold learning and semi-supervised learning," *Found. Trends Compt. Graph. Vis*., vol. 7, no. 2-3, pp. 81-227, 2012.

[21] F. Bossen, Common Test Conditions and Software Reference Configuration, document JCTVC-L1100, ITU-T/ISO/IEC Joint Collaborative Team on Video Coding (JCT-VC), 12th Meeting, Geneve, CH 14-23, Jan 2013.

[22] Joint Collaborative Team on Video Coding Reference Software, ver. HM 10.0 [Online]. Available: https://hevc.hhi.fraunhofer.de/ svn/svnHEVCSoftware/

[23] G. Bjøntegaard, "Calculation of average PSNR differences between RD Curves," in Proc. ITU-T SG16/Q.6 VCEG 13th Meet., Austin, TX, USA, Apr. 2001, Doc. VCEG-M33.

[24] G. Bjøntegaard, "Improvements of the BD-PSNR model," ITU-T SG16/Q.6 VCEG 35th Meet., Berlin, Germany, 2008. Doc. VCEG-AI11.

[25] M.U.K Khan, M. Shafique, J. Henkel, "An Adaptive Complexity Reduction Scheme with Fast Prediction Unit Decision for HEVC Intra Encoding," pp.1578-1582, Proceedings, *IEEE Intern. Confer. on Image Process. (ICIP2013)*, Melbourne, Australia, Sept. 2013.

[26] J. Kim, Y. Choe, and Y.-G. Kim, "Fast Coding Unit Size Decision Algorithm for Intra Coding in HEVC," pp.637-638, Proceedings, *IEEE Intern. Confer. on Consumer Electronics (2013ICCE)*, Berlin, Germany, Sept, 2013.