Block Partitioning Decision Based on Content Complexity for Future Video Coding

Yanhong Zhang[1,2](✉), Yao Zhao[1,2], Chunyu Lin[1,2], and Meiqin Liu[1,2]

[1] Institute of Information Science, Beijing Jiaotong University,
Beijing 100044, China
{17120332,yzhao,cylin,mqliu}@bjtu.edu.cn
[2] Beijing Key Laboratory of Modern Information Science and Network
Technology, Beijing 100044, China

Abstract. Recently, a block partition structure of quadtree plus binary tree (QTBT) has been proposed. Compared with the quadtree structure in HEVC, the block partition of QTBT is more flexible and the encoding performance is better, but at the same time, the encoding complexity is greatly increased. In order to better balance coding performance and complexity, we propose a block partitioning decision algorithm based on content complexity. By analyzing the variation of the complexity range of different splitting modes of adjacent frames, the unnecessary iteration process is reduced, thereby reducing the coding time complexity. The experimental results show that compared with the joint exploration test model (JEM), the average coding time of this method is reduced by 9.0%, while the coding performance is only lost by about 0.55%.

Keywords: Block partitioning · Quadtree plus binary tree · Content complexity · JEM

1 Introduction

The high-efficiency video coding standard HEVC/H.265 [1] is an international video compression coding standard proposed in 2013. In HEVC, video frames are first divided into equally-sized Coding Tree Units (CTUs), which are 64 × 64 in size. Then, the CTU is iteratively partitioned into coding units (CUs) according to a quadtree structure to adapt to different local features. Each CU can further divided into prediction units (PUs) and transform units (TUs). Although the HEVC structure greatly improves its performance over previous coding standards, it still has some problems.

To further optimize HEVC, the next-generation video coding standard H.266/VVC has been researched and developed. The Joint Exploration Test Model (JEM) is the test model for H.266. Among them, a new quadtree plus binary tree (QTBT) [2] structure was adopted by the Joint Video Experts Group (JVET) and integrated in JEM 3.0 and higher [3, 4]. In the QTBT structure, a more flexible CU partition type is supported. The size of the coding tree unit (CTU) is 128 × 128. CTUs are further divided into CUs, which are the basic units of encoding. Unlike HEVC, one CU in QTBT can be square or rectangular in shape. Figure 1 is an example of a CTU partition, with solid lines representing quadtree partitioning and dashed lines representing binary tree

Y. Zhao et al. (Eds.): ICIG 2019, LNCS 11903, pp. 70–80, 2019.
https://doi.org/10.1007/978-3-030-34113-8_7

partitioning. As can be seen from the CTU is first divided by a quadtree structure. Quadtree leaf nodes are further divided by a binary tree structure. There are two types in binary tree partitioning: symmetric horizontal partitioning and symmetric vertical. CU is not further divided into PUs and TUs. Therefore, the CU is also the basic unit of prediction and transformation.

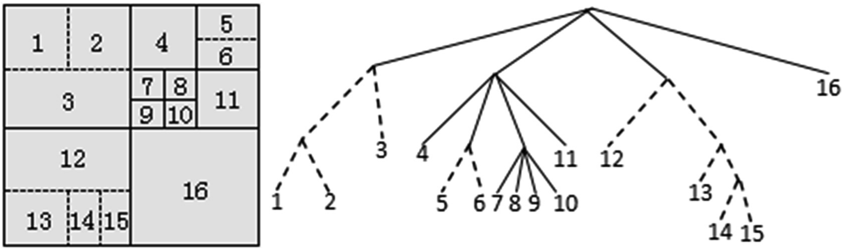

Fig. 1. Display of QTBT partition structure.

Due to the addition of the QTBT structure, in the JEM encoding, four types of partitioning attempts are required for each current block. They are unsplit, horizontal binary tree partitions, vertical binary tree partitions, and quadtree partitions. Among them, the type with the smallest RD cost is selected as the final division mode of the current block. The rate-distortion optimization process of JEM coding is shown in Fig. 2. As can be seen from the figure, when selecting the partition type for the current CU, in addition to the non-division mode, the other three partition types have to further recursively determine their own optimal partitions. The emergence of multiple partitioning structures allows CUs to be flexibly divided into different shapes to accommodate different video content. But it also leads to extremely high coding computation. Therefore, a fast algorithm needs to be proposed to reduce the consumption of coding time while ensuring stable coding performance.

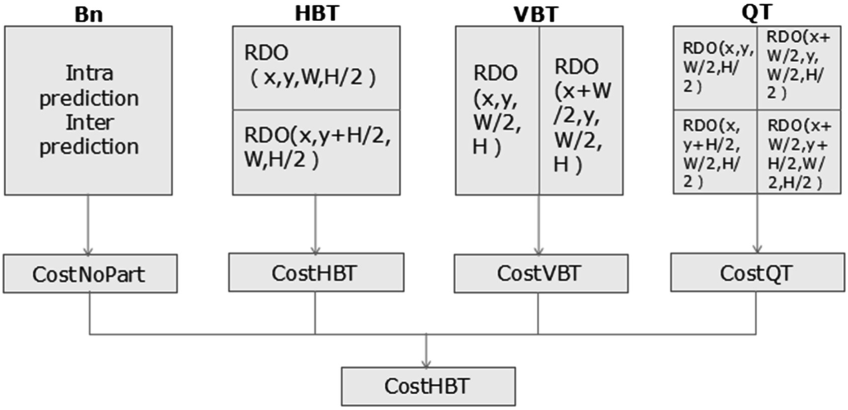

Fig. 2. Rate-distortion optimization process of JEM.

2 Related Work

After the above analysis, we know that the QTBT structure greatly improves the coding performance, but at the same time, the iterative process due to multiple partition types increases, which in turn leads to an increase in the encoding time. Therefore, an improved algorithm is needed to reduce the time complexity.

At this stage, many improvements have been proposed for algorithm acceleration. In [5], an algorithm is proposed to combine the CU coded bits with the reduction of unnecessary intra prediction modes to reduce the computational complexity. In [6], the author proposes a hybrid scheme consisting of a quick coding unit (CU) size decision and a fast prediction unit (PU) model decision process. In [7], a gradient-based intra-frame candidate mode clipping algorithm is proposed, which reduces the computational complexity by adaptive depth division and the use of spatial information to simplify the intra-frame prediction process. The above algorithm belongs to the traditional method. There are also some ways to use machine learning. In [8], the author proposes a HEVC inter-frame size decision algorithm. Several features that may be associated with the CU partition are selected by using an F-score based packaging method, and a three-output classifier is designed to control the risk of mispredictions by combining the classifier with the RD cost. In [9], an adaptive fast CU size decision algorithm is proposed. In this algorithm, firstly, the CU size decision process based on quadtree and the relationship between CU partition and image features are analyzed. Then, using Support Vector Machines, a three-output classification model is constructed based on CU complexity. Finally, the optimal CU size is predetermined by the model. The best CU size. In JEM, due to the appearance of the QTBT structure, the size and shape of the block division are different. Therefore, the above fast algorithm based on HEVC implementation cannot be directly applied to the QTBT structure.

Aiming at the QTBT structure, some improved algorithms are also proposed. In [10], a block segmentation technique based on probabilistic decision-making is proposed to identify unnecessary partition modes in terms of rate-distortion (RD) optimization. In [11], Wang et al. proposed an effective QTBT partition decision algorithm to achieve a good trade-off between computational complexity and coding performance. In [12], a fast intra-frame CU binary tree segmentation algorithm based on spatial features is proposed. By analyzing the different spatial features of the binary tree depth and the binary tree segmentation mode, the division of another binary tree is skipped directly.

The above proposed algorithms effectively reduce the coding complexity from different aspects, but these algorithms do not use the information of the complexity of adjacent frame content. We know that the content of video between adjacent frames is relatively similar. Therefore, we propose a fast partitioning algorithm based on content complexity.

The next part of the paper is organized as follows: In Sect. 3, block partitioning decision algorithm based on content complexity is presented. Experimental results and analysis are in Sect. 4. Section 5 is the summary of the paper.

3 Proposed Algorithm

In JEM coding, the partition size of a block is closely related to the complexity of the area to which the current block belongs. A region with a complex texture tends to split small blocks. Conversely, it tends to split large blocks. Then, the content of the image between adjacent frames does not change much, correspondingly, their partition structure is similar, which means that the content complexity and their splitting modes are similar between adjacent frames. In JEM coding, since the QTBT partition structure is added, the processing for the current block includes four cases. Therefore, it is desirable to reduce the coding time complexity by analyzing the variation of the complexity range of four different splitting modes of adjacent frames.

First, to get the complexity range of different partitioning methods, we calculate the complexity value of the current block first. Here, the standard deviation of the gray histogram of the current block is selected to represent its content complexity G. The specific calculation formula is as follows (1) and (2):

$$P_{average} = \frac{1}{H \times W} \sum_{i=1}^{H} \sum_{j=1}^{W} P_{i,j} \tag{1}$$

$$G = \frac{1}{H \times W} \sum_{i=1}^{H} \sum_{j=1}^{W} \left(|P_{i,j} - P_{average}| \right) \tag{2}$$

Where H and W represent the width and length of the current block, and $P_{i,j}$ represents the pixel value of the current block.

According to the above formula, we can get the content complexity of all CUs that select the same partition type in the same frame. The maximum and minimum values obtained constitute the complexity range of this kind of splitting. Here, the complexity of the four split methods is shown in Table 1:

Table 1. Complexity range representation of 4 partitioning methods.

(G_{MinBn}, G_{MaxBn})	The complexity range of all blocks that are not divided (Bn) in the same frame
(G_{MinHBT}, G_{MaxHBT})	The complexity range of all blocks that select horizontal binary tree partitioning (HBT) in the same frame
(G_{MinVBT}, G_{MaxVBT})	The complexity range of all blocks that select vertical binary tree partitioning (VBT) in the same frame
(G_{MinQT}, G_{MaxQT})	The complexity range of all blocks that select quadtree partitioning (QT) in the same frame

In order to oversee the variation of the complexity range of four different partitioning modes between adjacent frames of a video sequence, we calculate the complexity range of the BasketballPass video sequence. Table 2 shows the complexity range of four different partitioning modes in the first five frames of the sequence. From Table 2, we can see that the complexity range of selecting different partitioning modes is not exactly the same in the same frame. At the same time, the complexity range of selecting the same splitting method is similar between adjacent frames. Therefore, we can effectively reduce the coding complexity based on these two characteristics.

First, we can encode the first frame according to the original encoding process and obtain the complexity range corresponding to different partition modes. This is shown in Fig. 3. Among them, the shade of the color in the figure indicates the number of split modes that need to be tried. From deep to shallow, there are four, three, two, and one. In the next frame coding process, the unnecessary partitioning mode is directly skipped by judging the scope of the current block complexity. For example, the complexity value of the current block is 160. Since it is in the range of 158 to 165, it is only necessary to perform three partitioning attempts on the current block without splitting by the horizontal binary tree. If the complexity value is greater than 165, then only one of the partitioning modes of the current block can be tried, thereby reducing the time consumption caused by the other three partitions.

Table 2. Complexity range G of 4 divisions of the first 5 frames of BasketballPass.

	Bn	HBT	VBT	QT
Frame 1	(4,165)	(6,158)	(8,165)	(10,524288)
Frame 2	(3,166)	(8,149)	(10,158)	(18,524288)
Frame 3	(4,165)	(8,143)	(10,157)	(18,524288)
Frame 4	(4,157)	(8,138)	(8,157)	(18,524288)
Frame 5	(4,165)	(6,147)	(7,157)	(18,524288)

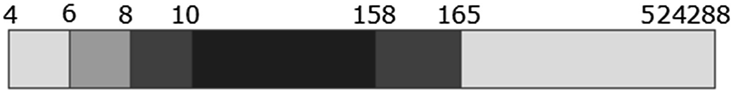

Fig. 3. The number of split modes that need to be tried corresponding to different complexity ranges of the first frame of BasketballPass (Colors from deep to shallow indicate that it need to try four, three, two and one.).

Through the above analysis, we know that due to the content correlation of adjacent video frames, the time consumption caused by unnecessary iterations can be reduced according to the complexity range of different partitioning modes. However, the above-mentioned complexity range does not consider different depth cases. The partitioning of the current block is closely related to its depth. For example, as the depth increases, the partitioning of the quadtree may be less used. Therefore, in order to make block partitioning decisions more accurately, the quadtree plus binary tree depth (uiQTBT-Depth) is taken into consideration. In other words, the complexity values calculated by selecting the same partitioning mode in one frame are separately counted according to different depths. Table 3 is a representation of the complexity range of four different partitioning modes with a depth of 2 in one frame of the BasketballPass video sequence. We compare the two situations with or without considering depth Fig. 4. Figure 4(a) is a complexity range when depth is not distinguished. When the depth is 2, we display the range of 10 to 158 according to the data in Table 3, as shown in Fig. 4 (b). It can be seen from the figure, for the complexity value calculated by the current block, in two cases, the required number of partitioning iterations may be different. For example, the content complexity of the current block is 48. When the depth is not distinguished, the four partitioning modes need to be tried once. When the depth is distinguished, only two partitioning attempts are required.

Table 3. The complexity range of different partitioning modes when the first frame of BasketballPass Depth = 2.

	Bn	HBT	VBT	QT
Depth = 2	(0,0)	(22,46)	(26,66)	(22,524288)

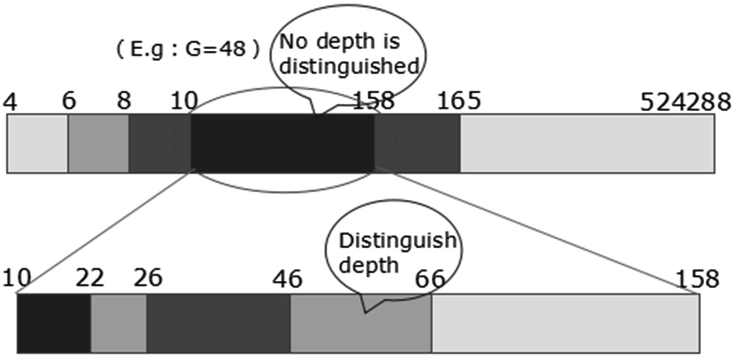

Fig. 4. Display of the number of divisions of the same complexity value in two cases. (a) Depth considered. (b) Without regard to depth.

Based on the above analysis, this paper proposes a partitioning decision algorithm based on content complexity. The overall process is shown in Fig. 5. In this algorithm, by counting the complexity ranges of different depths in the first frame, unnecessary partitioning attempts are reduced, thereby reducing coding time complexity. First, for each current block, its complexity *G* is calculated and its current partition depth *d* is obtained. If the current block belongs to the first frame, encode according to the original encoding process and update the corresponding complexity range. For other frames, if the current frame is entered for the first time, the complexity range corresponding to the depth *d* in the previous frame is obtained. Otherwise, the complexity range of the current frame depth *d* is obtained. Next, the unnecessary partitioning process is eliminated according to the range to which *G* belongs. Of course, if *G* is not within any complexity range, then the original encoding process is still performed. Similarly, in order to make the subsequent block partitioning more precise and efficient, the *G* value of the current block is also updated according to its corresponding complexity range.

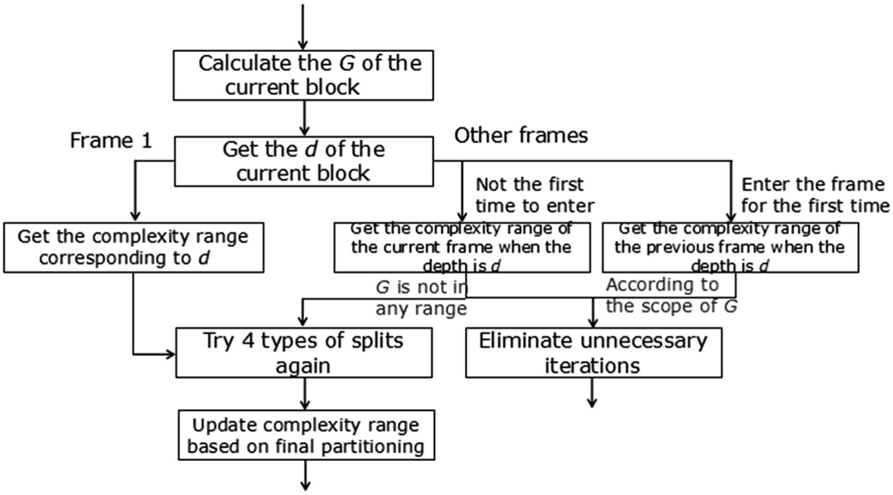
Calculate the G of the current block
Frame 1
Get the d of the current block
Other frames
Not the first time to enter
Enter the frame for the first time
Get the complexity range corresponding to d
Get the complexity range of the current frame when the depth is d
Get the complexity range of the previous frame when the depth is d
G is not in any range
According to the scope of G
Try 4 types of splits again
Eliminate unnecessary iterations
Update complexity range based on final partitioning

Fig. 5. Overall algorithm work process.

4 Experimental Results

To verify the performance and efficiency of the proposed algorithm, we performed the following experiments. We integrated it into the reference software HM-16.6-JEM-4.2 released by JVET. The test video sequence used is a common test sequence recommended by JVET. The selection of video sequences involves multiple categories to ensure the accuracy of test results. All the experiment was performed in both lowdelay and random access configurations and used four different QPs (22, 27, 32, 37). The evaluation criteria used for this experiment were BD-Rate and ΔET. Among them, BD-Rate represents the reduction of bit rate when the peak signal-to-noise ratio between the anchor and the algorithm is equal. ΔET is compared with the anchor, the algorithm reduces the time ratio as the formula (3) shown. Where T_{JEM} represents the time spent under the JEM source code and $T_{\mathrm{Pr}op}$ represents the time used in the method.

$$\Delta ET = \frac{1}{4}\sum_{i=1}^{4} \frac{T_{JEM} - T_{\mathrm{Pr}op}}{T_{JEM}} \times 100\% \tag{3}$$

Compared with the JEM encoding process, the performance evaluation are shown in Table 4. Positive value of BD-Rate indicates a decrease in coding performance, and negative value of ΔET indicates a decrease in coding time. Under the Random Access configuration, the encoding time is reduced by an average of 8.3%, and the loss of encoding performance is only 0.2%. In the Lowdelay configuration, the encoding time is reduced by an average of 9.5%, and the loss of encoding performance is only 0.89%. From the data in the table, we can also see that the sequence RaceHorses can save more time than the sequence BQSquare and FourPeople. This is because the texture is complex in RaceHorses. Correspondingly, it needs to divide more blocks, so the time saved by the proposed algorithm is more. At the same time, because of its intense motion, the deviation between adjacent frames is slightly larger than other sequences, so the performance degradation is relatively large. For sequences with a simple background and slow motion, such as Kimono, performance degradation is negligible. This is because the slow motion makes the similarity between adjacent frames extremely high, and the resulting block partition structure is more accurate.

Table 4. Performance of the proposed algorithm.

Sequence	Proposed random access BD-rate	ΔET	Lowdelay BD-rate	ΔET
Cactus	+0.51%	−6.58%	+0.72%	−5.86%
Kimono	+0.10%	−4.91%	−0.08%	−5.10%
BQMall	+0.14%	−8.32%	+1.29%	−7.09%
BasketballDrill	+0.16%	−9.56%	+2.72%	−24.92%
BQSquare	+0.04%	−6.36%	+1.45%	−10.21%
RaceHorses	+1.02%	−17.30%	+1.04%	−16.75%
BasketballPass	−0.01%	−11.09%	+0.70%	−12.98%
FourPeople	+0.15%	−2.97%	+0.01%	−2.46%
KristenAndSara	−0.31%	−7.59%	+0.16%	−0.17%
Average	+0.20%	−8.30%	+0.89%	−9.50%

Figure 6 shows the rate-distortion comparison between the algorithm and JEM encoding in the RaceHorses sequence in two different configurations. The figure shows the rate savings for the two methods at the same objective quality, and the difference in PSNR-Y between the two methods at the same code rate. It can be seen from the figure that the performance difference between the algorithm and the original JEM encoding is not large, and the performance loss can be neglected.

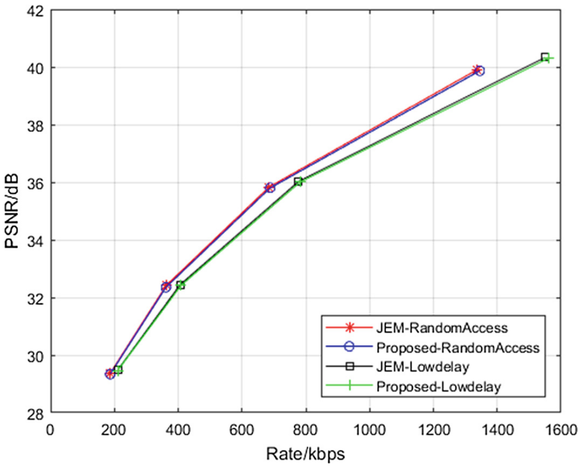

Fig. 6. The rate distortion curve of RaceHorses in both configurations.

Figure 7 shows the number of iterations reduced in frames 2 through 8 of the RaceHorses video sequence, respectively. As can be seen from the figure, the method can reduce the number of iterations of hundreds or even thousands for each frame. At the same time, we can also see that the later the frame to be encoded, the more the number of iterations is reduced. This is because, as the block is divided, the later the current frame is, the more information it can refer to.

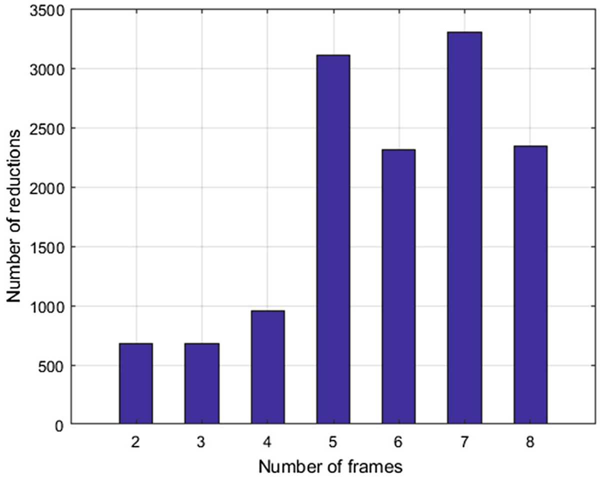

Fig. 7. The number of iterations reduced in the 2nd to 8th frames of RaceHorses.

In order to more objectively demonstrate the impact of the algorithm on the block partition structure, the number of different partition types for the same size CU is demonstrated for the JEM and the proposed algorithm. The statistical result of all

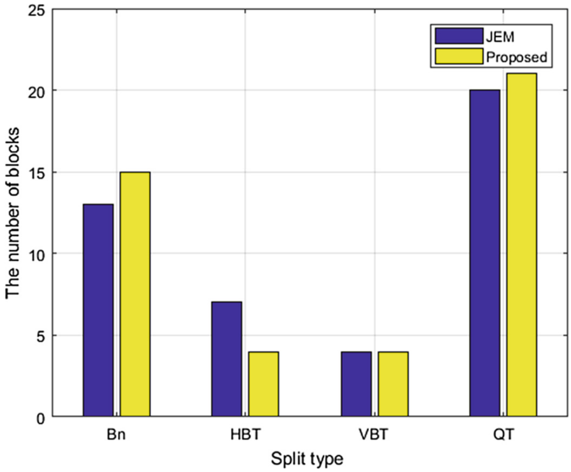

Fig. 8. Number of different partition types of 64 × 64 blocks in the 5th frame of BasketballPass.

64 × 64 sized blocks in the 5th frame of the BasketballPass sequence is shown in Fig. 8. It can be seen from the figure that the structure of block partitioning is similar in the two modes. Only a few of the blocks are selected in different types. Therefore, it is further proved that the current fast algorithm saves time while the selected mode is basically the same as JEM.

5 Conclusion

In this paper, we propose a block partitioning decision algorithm based on content complexity that is used to reduce encoding complexity while ensuring encoding performance. By analyzing the relationship between the content complexity of the video content and the four partition methods, the partial splitting mode attempt is terminated in advance, thus the coding complexity is achieved. Experimental results show that the average encoding time of the algorithm is reduced by 9.0%, while the coding performance loss is about 0.55%. This method achieves a good balance of coding performance and complexity.

Acknowledgment. This work was supported in part by the Fundamental Research Funds for the Central Universities (2018JBZ001), and by National Natural Science Foundation of China (No. 61772066).

References

1. Sullivan, G.J.: Overview of the high efficiency video coding (HEVC) standard. IEEE Trans. Circ. Syst. Video Technol. **22**(12), 1649–1668 (2012)
2. An, J.: Quadtree plus binary tree structure integration with JEM tools. JVET-B0023. In: Joint Video Exploration Team (JVET), San Diego, USA (2016)
3. Li, X.: JEM software development (AHG3). Document JVET-J0003-v1. In: Joint Video Exploration Team (JVET), San Diego, USA (2018)
4. Alshina, E.: Performance of JEM 1 tools analysis. Document JVET-B0022. In: Joint Video Exploration Team (JVET), San Diego, USA (2016)
5. Zhang, M.: Fast algorithm for HEVC intra prediction based on adaptive mode decision and early termination of CU partition. In: 2018 Data Compression Conference, Snowbird, UT, USA, p. 434 (2018)
6. Lu X.: A fast HEVC intra-coding algorithm based on texture homogeneity and spatio-temporal correlation. EURASIP J. Adv. Sig. Process. 1–14 (2018)
7. Shi, Z.: Gradient-based and intra-frame adaptive depth decision algorithm. J. Shanghai Normal Univ. (Nat. Sci.) **47**, 248–252 (2018)
8. Gao, X.: A fast HEVC inter CU size decision algorithm based on multi-class learning. In: 2018 10th International Conference on Intelligent Human-Machine Systems and Cybernetics (IHMSC), Hangzhou, China, pp. 64–68 (2018)
9. Liu, X.: An adaptive CU size decision algorithm for HEVC intra prediction based on complexity classification using machine learning. IEEE Trans. Circ. Syst. Video Technol. **29**(1), 144–155 (2019)
10. Wang, Z.: Probabilistic decision based block partitioning for future video coding. IEEE Trans. Image Process. **27**(3), 1475–1486 (2018)

11. Wang, Z.: Effective quadtree plus binary tree block partition decision for future video coding. In: 2017 Data Compression Conference, Snowbird, UT, USA, pp. 27–32 (2017)
12. Lin, T.: Fast binary tree partition decision in H.266/FVC intra coding. In: IEEE International Conference on Consumer Electronics, Taichung, Taiwan, pp. 1–2 (2018)