IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received 2 July 2022, accepted 16 July 2022, date of publication 25 July 2022, date of current version 28 July 2022.

Digital Object Identifier 10.1109/ACCESS.2022.3193401

RESEARCH ARTICLE

High Efficiency Intra CU Partition and Mode Decision Method for VVC

CHI-TING NI, SHIH-HSIANG LIN, PEI-YIN CHEN, (Senior Member, IEEE), AND YU-TING CHU

Digital IC Design Laboratory, Department of Computer Science and Information Engineering, National Cheng Kung University, Tainan 70101, Taiwan

Corresponding author: Pei-Yin Chen (pychen@csie.ncku.edu.tw)

This work was supported in part by the Ministry of Science and Technology, Taiwan, under Grant 108-2221-E-006 -096-MY3.

ABSTRACT Versatile video coding (VVC/H.266) is the newest video compression standard, which is developed by the Joint Video Experts Team. Compared with previous encoding schemes, VVC achieves higher compression efficiency by introducing a new partition structure and additional intra prediction modes but results in high computational complexity. To efficiently solve this problem of redundant processing in quad-tree with nested multi-type tree structures and intra mode prediction, we propose a texture analysis–based ternary tree (TT) and binary tree (BT) partition strategy, and a gradient-based intra mode decision method to accelerate TT and BT partition and intra mode prediction, separately. The texture complexity and prediction direction of coding unit (CU) is calculated by texture detection method. A texture analysis–based TT and BT partition strategy is established by using the regression method based on analyzing the texture complexity of the CU. Then, a texture analysis–based TT and BT partition strategy is applied to reduce the redundant partition for each CU. By using the prediction direction of CU, a gradient-based intra mode decision method is established for skipping the impossible modes for each CU. Experimental results revealed that the proposed method could save 49.49% in encoding time and increase the Bjontegaard delta bit rate (BDBR) by only 0.56%. It confirms that the proposed method achieved high efficiency and a good balance between the BDBR and time saving.

INDEX TERMS H.266/VVC, fast CU partition, intra mode decision, gradient-based, regression.

I. INTRODUCTION

With the rapid development of video applications, high-resolution video, such as 4K or 8K ultra high definition video, is increasingly popular. The huge amount of data has become a new challenge for data transmission. The advanced high-efficiency video coding standard (HEVC/H.265) [1] is unlikely to meet the increasing market demands in the future. Therefore, a new encoding scheme must be developed to achieve higher compression efficiency. Accordingly, versatile video coding (VVC/H.266) [2] was developed by the Joint Video Experts Team, which consists of the ITU-T Video Coding Experts Group and ISO/IEC Moving Picture Experts Group. VVC is a new video compression standard that offers greater compression efficiency than HEVC, which was finalized in July 2020 [2].

The associate editor coordinating the review of this manuscript and approving it for publication was Chong Leong Gan.

Many new encoding technologies [3] have been adopted in VVC. For example, quad-tree with nested multi-type tree (QTMT) has been employed in the coding tree unit (CTU) partition process, 67 intra modes with wide-angle mode extension has been adopted in intra prediction, and affine motion inter prediction has been adopted in inter picture prediction. VVC is a block-based hybrid coding architecture. Each frame of the input video is divided into many blocks called CTUs, and these CTUs are divided into many smaller blocks called CUs.

In VVC, a CTU is first partitioned by using a quad-tree structure as in HEVC, and then the leaf nodes of the quad-tree can be split by using a nested multi-type tree (MTT) to ensure that the CUs in VVC are more flexible and more suitable for the intra and inter prediction. Fig. 1 illustrates a CTU partitioned by a QTMT structure. Only the leaf node of the QT split can be partitioned with an MTT structure. MTT structures commonly involve four splitting modes: horizontal

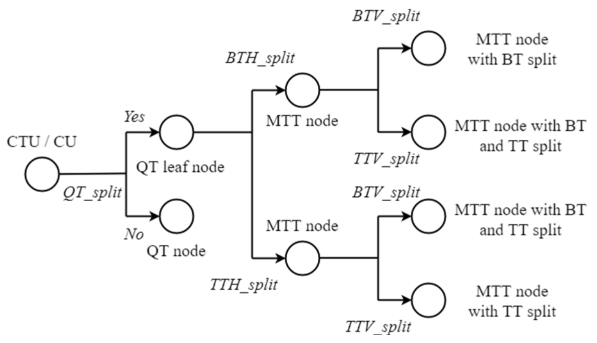


FIGURE 1. Overview of the proposed disparity estimation algorithm.

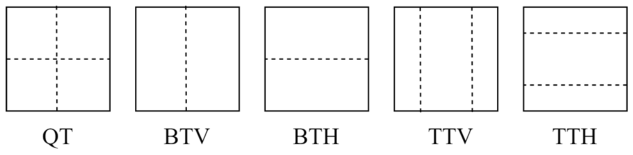


FIGURE 2. Splitting modes used in QTMT structures.

binary tree partition (BTH), vertical binary tree partition (BTV), horizontal ternary tree partition (TTH), and vertical ternary tree partition (TTV), as depicted in Fig. 2.

By using these technologies, VVC has approximately 50% higher coding efficiency than HEVC. However, these new technologies also sharply increase computational complexity in both the encoder and decoder. For example, the QTMT structure can more closely fit the block texture than HEVC but results in higher coding computational complexity [4]. After a CTU is divided into several CUs, those CUs are used in intra prediction to select the optimal intra mode from 67 intra prediction modes. The intra prediction of VVC is based on the three steps described in [5] and [6]. In the final step, all CUs perform rate distortion optimization (RDO) [7] and select the optimal intra mode according to the minimal rate distortion cost (RD-cost). The RD-cost is expressed as (1), where RD_{cost} is the RD-cost, λ is the Lagrange multiplier, R is the number of encoding bits and D is the reconstruction distortion. Although the extended modes and QTMT structure can improve compression efficiency, the encoding complexity increases significantly. Hence, an acceleration algorithm is proposed to solve the problem.

$$RD_{cost} = D + \lambda \times R \tag{1}$$

In this study, we first analyzed the complexity of the intra prediction with different CU sizes by using various video sequences. Then, we provided TT and BT partition strategies based on texture analysis and the gradient-based intra mode decision method to reduce computational complexity. Experimental results revealed that the proposed method achieved high efficiency and a good balance between the BDBR [8] and time saving by using the various video sequences in different version of the VVC test model (VTM).

II. RELATED WORK

Over the years, several studies have been devoted to accelerating the intra prediction or CU partition no matter based on H.266/VVC, H.265/HEVC, H.264/AVC, or other coding standards. Several studies have presented texture feature– or machine learning (ML)-based techniques to reduce the redundancy for HEVC involved with intra prediction in HEVC [9]–[13]. Compared with HEVC, VVC introduced many new techniques to obtain higher video compression performance and more flexible coding. For example, VVC adopts the QTMT structure to split the CUs and uses 67 modes in intra prediction instead of the 35 modes used in HEVC; these increase the compression efficiency compared with HEVC, but these also increase the encoding complexity significantly. To reduce the encoding complexity, many methods have been devised. Other works have focused on speeding up the encoders by reducing the redundancy process of the partition [14]–[18]. A convolutional neural network was applied to reduce the QTBT partitioning in [14]–[16]. Wang *et al.* [17] used a joint classifier decision tree structure to eliminate unnecessary iterations that increase the computational complexity. Amestoy *et al.* [18] used random forest classifiers to estimate the optimal partition mode for each coding block. In addition to the QTBT structure, Amestoy used random forest classifiers to deal with MTT partitioning. Because VVC has integrated the MTT structure, many methods have recently been proposed to reduce the coding complexity of VVC by using the MTT structure.

A low-complexity, statistical learning–based CTU partition structure decision and gradient-based fast intra mode decision for VVC was proposed in [4], which pioneered the study of the acceleration algorithm for use in the MTT structure, yielding experimental results with the proposed method that demonstrated a favorable balance between the BDBR and time saving. Although the method proposed in [4] is notable, its VTM version is outdated, and the BDBR performance on some test sequences is suboptimal. Some researchers are currently working to improve the performance of the newest version.

A variance- and gradient-based fast intra partition algorithm was proposed in [19] to speed up the CU partition. Zhang *et al.* [20] designed a CU partition (on the basis of a random forest classifier model) and an intra mode decision algorithm (on the basis of texture region features). Zhang *et al.* [21] proposed a CU size decision method for intra prediction of VVC according to texture complexity, which can reduce the computational complexity of VVC's intra encoding. A variance- and Sobel operator–based fast intra partition algorithm was proposed by [22] to decide whether to split a given CU by using QT and thus terminate further MTT partitions. Park and Kang [23] proposed a simple early decision technique that can effectively reduce TT complexity. Li *et al.* [24] proposed a tunable decision model based on the prediction distortion to achieve different trade-off between encoding loss and complexity reduction.

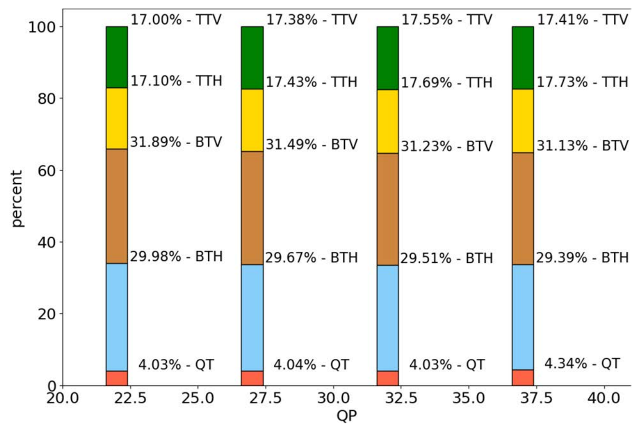


FIGURE 3. Time distributions of different splitting modes obtained using QP 22, 27, 32, 37.

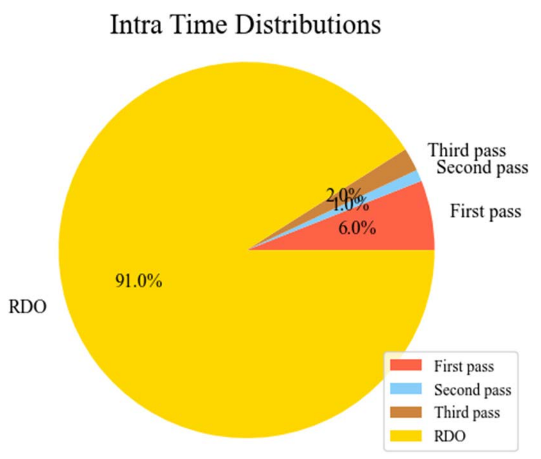


FIGURE 4. Time distributions of intra prediction with QP 22, 27, 32, 37. First pass: select M mode from 35 modes, second pass: add neighboring mode from M modes, third pass: merge the modes which derived from neighboring CUs, and RDO: select the optimal mode from the final list.

Although these methods can accelerate the intra prediction process in VVC, the version of the VTM used is outdated, and some new techniques for VVC are not integrated. Furthermore, in these existing works, the correlation between intra prediction and texture features is not well utilized. Therefore, there is still a lot of potential to improve the trade-off between complexity reduction and encoding loss.

III. PROPOSED METHOD

The original encoding process in the VVC/H.266 selects the optimal partition scheme with the lowest RD-cost in intra prediction of all the possible CUs. Because each CU has a variety of splitting modes and each possible CU must execute intra prediction, considerable time is required to determine the optimal partition scheme. To reduce the encoding complexity, we analyzed the process of intra prediction and determined the most complex part. According to the analysis results, high-efficiency algorithms were designed for intra mode prediction and CU partition, separately. In the following section, details of the analysis and the proposed algorithm are provided.

A. COMPLEXITY ANALYSIS OF CU PARTITION AND INTRA PREDICTION

The simulation conditions follow the all-intra mode specified in the common test conditions of the JVET [25] for the VTM. The test was adapted from the standard test sequences with quantization parameters (QPs) $\in$ 22, 27, 32, 37. Fig. 3 illustrates the time distributions for MTT partitioning with different QPs. Among the splitting modes, QT splitting took the least time regardless of which QP was employed. The TT split portion took approximately 33% the overall encoding time, and the BT split portion accounted for nearly 60% with each QP.

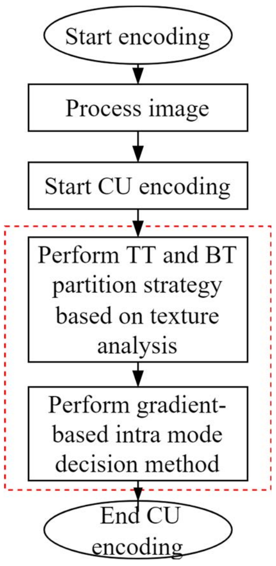
Regardless of the QP, the BT and TT structures were more complex than the QT structure; hence, determining how to reduce the complexity of BT and TT partitioning is necessary. Accordingly, we proposed the TT and BT partition strategy based on texture analysis to reduce the number of redundant iterations.

With the same test environment as that used in the aforementioned time distribution analysis, the time distributions of the intra prediction are presented in Fig. 4. The first pass involves selecting the optimal M from among the 35 modes. The second pass adds the neighboring modes of the M modes and then selects the optimal N modes into a rough mode decision (RMD) mode list. The third pass then merges the modes derived from neighboring CUs into the RMD mode list and establishes a final candidate list. RDO is employed to select the optimal mode from the final candidate list. Among these operations, RDO takes the longest time to obtain the optimal result. To reduce the time required (i.e., number of modes calculated) for the RDO, a gradient-based intra mode decision method was devised.

The proposed algorithm consists of two strategies, namely TT and BT partition strategy based on texture analysis and gradient-based intra mode decision method. The proposed algorithm flow is depicted in Fig. 5. Notably, the proposed algorithm replaces the parts of CU partition and intra mode prediction. In the following section, details of the aforementioned TT and BT partition strategies are provided.

B. TT AND BT PARTITION STRATEGY BASED ON TEXTURE ANALYSIS

The texture analysis–based TT and BT partition strategies have two parts of block texture detection and overlapping coverage detection. Block texture detection is applied first, and then the results are used to determine whether the current CU can cease early. After the BT partition, the CU will apply overlapping coverage detection to skip the unnecessary TT partition. The algorithm flow is depicted in Fig. 6. In the following section, details of the block texture detection and overlapping coverage detection are provided.



Start encoding
Process image
Start CU encoding
Perform TT and BT partition strategy based on texture analysis
Perform gradient-based intra mode decision method
End CU encoding

FIGURE 5. Flow for the algorithm proposed in this study.

1) BLOCK TEXTURE DETECTION

Generally, the gradient of a block can be used to assess whether the current block is homogeneous or not. For example, if the block contains more textures, the edge of the block is more obvious and the gradient is larger, and vice versa. In addition to the gradients, the directions of the texture in the block are required because the gradient-based intra mode decision method involves using those directions to determine the main direction of the CU. To obtain the gradient and the direction, the edge detection operator is applied to the current CU to process the luma component.

First, the gradient of the current frame is computed before the CU partition. G_x and G_y are used to calculate the gradient G of the current frame. G_x and G_y are the gradients in the horizontal and vertical directions, respectively. Thus, G_x and G_y are extracted by using the edge detection operator, expressed as (2) and (3), respectively. The $P_{i,j}$ in (2) and (3) is the 3×3 intensity matrix in the current frame and is centered on the point currently being computed. i and j represent the position of the current center pixel in a row and in a column, respectively. The gradient G is the square root of G_x^2 and G_y^2, and the equation is expressed as in (4).

$$G_x(i,j) = P_{i,j} \times \begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix} \tag{2}$$

$$G_y(i,j) = P_{i,j} \times \begin{bmatrix} 1 & 2 & 1 \\ 0 & 0 & 0 \\ -1 & -2 & -1 \end{bmatrix} \tag{3}$$

$$G(i,j) = \sqrt{G_x(i,j)^2 + G_y(i,j)^2} \tag{4}$$

Before adopting the edge detection operator, the current frame is padded first to prevent a size change. After padding, for the current frame, (2), (3), and (4) are computed and the results of G_x, G_y and G in every pixel of the current

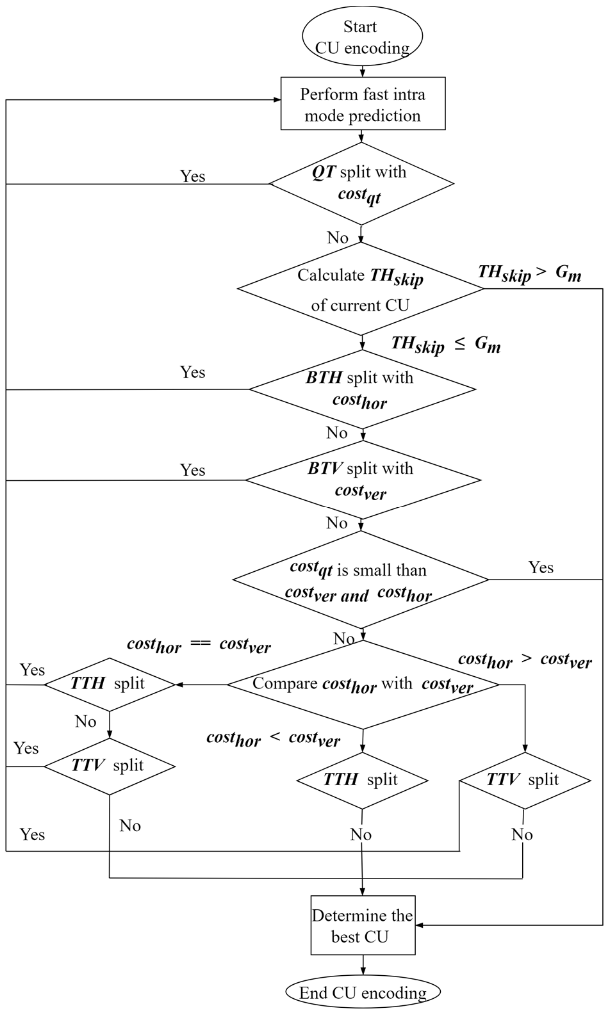


FIGURE 6. Algorithm flow for TT and BT partition strategy based on texture analysis.

frame are stored. Eventually, G_x, G_y, and G in every pixel of the current frame are computed, and the results are used to determine whether the current CU requires splitting. The maximum G of the CU can be used to assess whether the block is flat. The smaller the magnitude of the gradient is, the flatter the current CU is. The greater the magnitude of the gradient is, the more complex the current CU is. The current CU being flat implies that splitting can be terminated early. The relationship between the maximum G of the CU and split mode is illustrated in Fig. 7. In Fig. 7(a), the maximum G of the CU is 55 and that in Fig. 7(b) is 2920. The splitting results displayed in Fig. 7(a) and 7(b) are no-split and TT split, respectively. It can be clearly seen from Fig. 7, if the maximum G of the CU is too small, the result of the split mode tends to be no-split. Therefore, the gradient can be used to assess whether the current block is flat. However, the factors that determine the partition mode relate not only to the gradient but also to the area of the CU and QP. In order to study the influence of gradient, area and QP on the partition mode decision in the video encoding process,

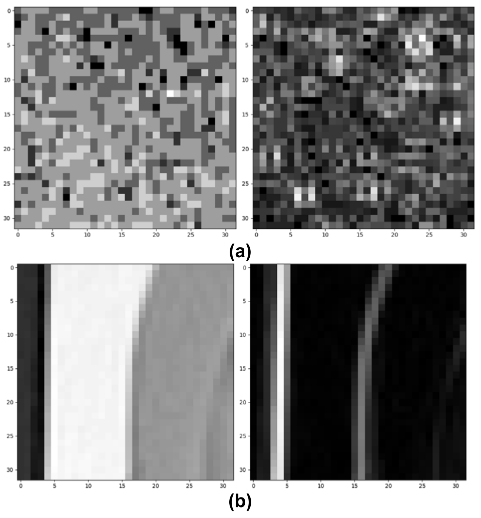


FIGURE 7. **Relationship between the gradient and splitting mode. The maximum magnitude values for the cases shown in (a) and (b) are small and large, respectively.**

we introduced the *average no − split ratio* (NP_a) of the CU generated during the encoding of several video sequences. NP_a is calculated by (5), (6), and (7), where the G_m is the maximum magnitude of the gradient in the current CU, $split_{mode}$ is the partition mode of the CU, th is different value that wants to compare with G_m, NUM_{np} is the total no-split number in the dataset, and N is the number of the CU in the dataset. The dataset consists of several sequences, where the sequences include Keiba, Mobisode2, Flowervase, vidyo1, SteamLocomotiveTrain, and PeopleOnStreet. We take the former 30 frames of each video to study the relationship and build the regression function.

$$NP\left(G_m, th\right) = \begin{cases} 1, G_m < th \cap split_{mode} \in no\ split \\ 0, else \end{cases} \tag{5}$$

$$NP_n = \sum_{n=1}^{N} NP(G_m, th) \tag{6}$$

$$NP_a = \frac{NP_n}{NUM_{np}} \tag{7}$$

$$A_{CU} = CU_w \times CU_h \tag{8}$$

Fig. 8 illustrates the relationship between the G_m of the CU and various QPs with a fixed area by using various th. It can be seen that the higher the QP is, the lower the no-split ratio is at a given th. This means that if it wants to achieve same no-split ratio at a higher QP, the higher th must be obtained because a no-split result is more likely to be determined for the CU with the higher QP. The lower the QP is, the more details of the frame can be saved, yielding higher video quality but also a higher bit rate. Conversely, the higher the QP is, the more details of the frame are skipped, which leads to lower video quality but also a lower bit rate.

Fig. 9 displays the correlation between the G_m of the CU and various A_{CU} values with a fixed QP by using various th. The A_{CU} is calculated according to (8), where CU_w is the width of the current CU and CU_h is the height of the current CU. The larger the A_{CU} is, the larger the no-split ratio is at the same threshold, suggesting that if the A_{CU} is larger, the CU is more likely to be determined to involve a partition mode with the same threshold. In order to express the correlation between the G_m, QP, and A_{CU} as an equation and insure that the CU needs to split can be judged, we introduced the *average split ratio* (SP_a) and various SP_a are used for testing to get the best trade-off between BDBR and time saving.

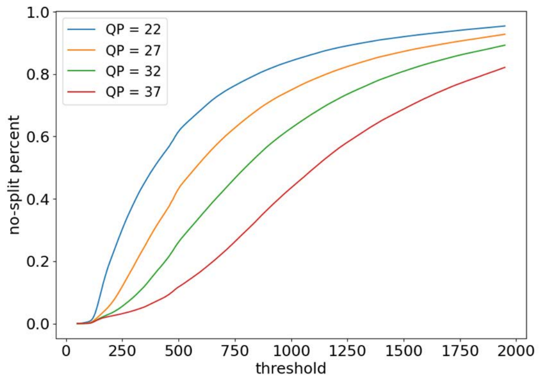


FIGURE 8. **No-split distributions of various QPs. The x-axis represents the different value of th and the y-axis represents the different value of NP_a.**

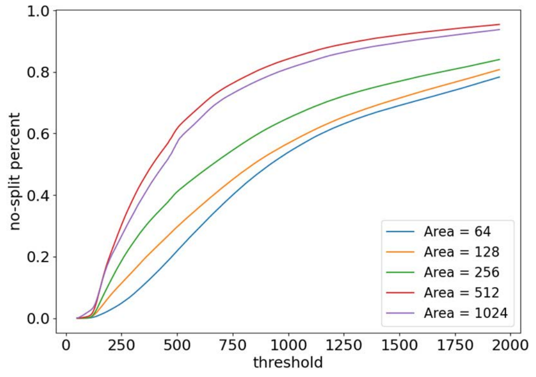


FIGURE 9. **No-split distributions of various area. The x-axis represents the different value of th and the y-axis represents the different value of NP_a.**

$$SP\left(G_m, th\right) = \begin{cases} 1, & G_m \geq th \cap split_{mode} \in BT\ or\ TT\ split \\ 0, & else \end{cases} \tag{9}$$

$$SP_n = \sum_{n=1}^{N} SP(G_m, th) \tag{10}$$

$$SP_a = \frac{SP_n}{NUM_{sp}} \tag{11}$$

SP_a is calculated by (9), (10), and (11), where the G_m is the maximum magnitude of the gradient in the current CU, $split_{mode}$ is the current partition mode of the CU, th is different value that wants to compare with G_m, NUM_{sp} is the total split number in the dataset, and N is the number of the CU in the dataset. The composition of the dataset is the same as previous.

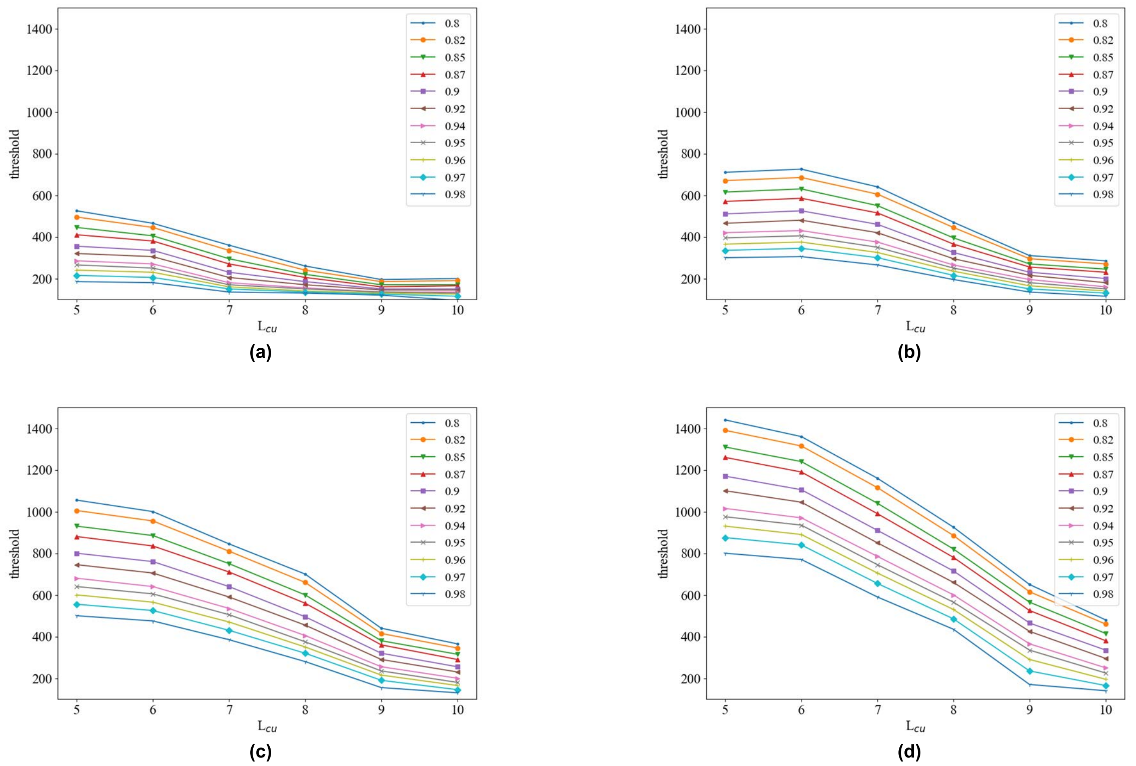


FIGURE 10. Distributions for different split ratio obtained with (a) QP22, (b) QP27, (c) QP32, and (d) QP37. The x- and y-axis represent the L_{CU} and threshold, respectively.

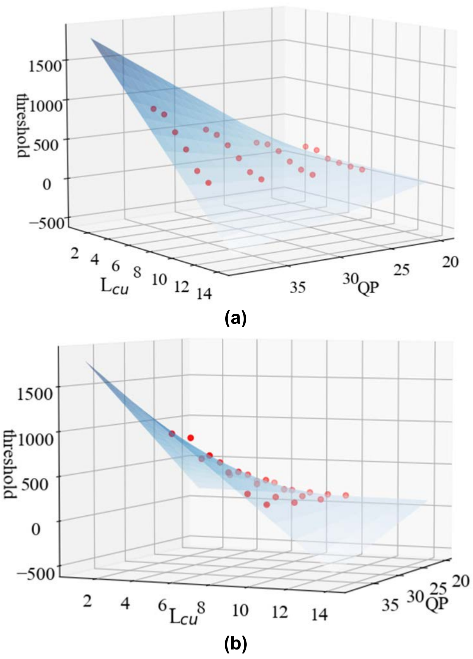


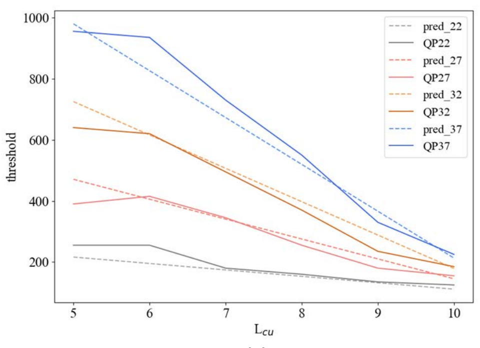
FIGURE 11. The surface for fixed split ratio. The red points are the ground truth and the blue surface is predicted surface. (a) and (b) are the pictures from different angles.

Fig. 10 shows the distributions for split ratio with distinct QPs and L_{CU} sizes, where L_{CU} is calculated according to (12). Because the differences in values among the various A_{CU} are excessively large and nonlinear, (12) is applied to A_{CU}. The trends of the distributions for different split ratio are consistency; therefore, a polynomial can be used to express the distributions for distinct SP_a with different coefficients. The threshold can be obtained quickly and suitable for hardware implementation, so that we adopted the linear polynomial to calculate the threshold. The optimal linear polynomial is obtained by regression method, and it is expressed as (13), where a, b, c and d are constants with different SP_a and TH_{skip} is used to decide if the present CU can be skipped or not. Fig. 11 and Fig. 12 illustrates the correlation between the surface calculated in (13) and points of the fixed SP_a. The error between the points and the surface is only slight, therefore, the accuracy of SP_a can be maintained by using TH_{skip}. Fig. 13 shows the relationship between the split ratio, BDBR, and time saving, tested according to various video sequences. Notably, the optimal balance between the BDBR and time saving is achieved when the split ratio is set to 0.94.

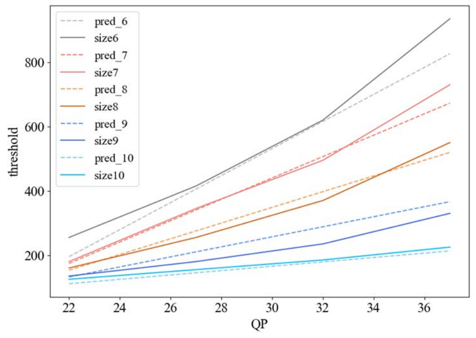
$$L_{CU} = log_2(A_{CU}) \tag{12}$$

$$TH_{skip} = a \times QP + b \times L_{CU} + c \times QP \times L_{CU} + d \tag{13}$$

Before the BT and TT split of the CU partition, the G_m is obtained in the current CU and is used to assess whether the current CU is homogeneous. As mentioned, the correlation between the G_m, QPs, and L_{CU} is calculated according



(a)



(b)

FIGURE 12. The comparison of predict value with ground truth. (a) is various QPs with the different size, where the size is calculated by (12). (b) is various size with the different QPs. The solid line is ground truth and the dashed line is predict values.

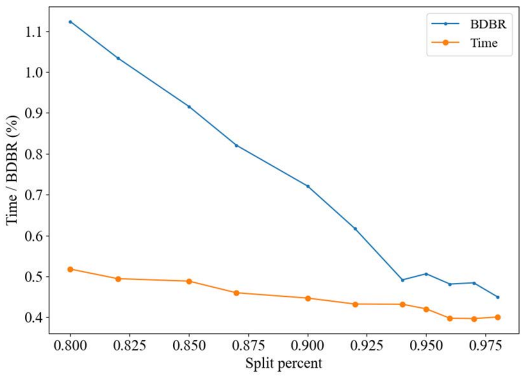


FIGURE 13. The relationship between BDBR, time saving and split ratio.

to (13). The skip condition of the block texture detection is that if the G_m is smaller than the threshold TH_{skip}, the current CU will skip the splitting modes of BTH, BTV, TTH, and TTV. If G_m is greater than or equal to the threshold TH_{skip}, the current CU performs BTH and BTV. Overlapping coverage detection is introduced and how to reduce the TT split is explained in the following section.

2) Overlapping coverage detection: Park and Kang [23] analyzed the MTT structure and discovered that the complexity of TT partition is higher than that of BT partition. Our proposed method uses the concepts of [23] to skip the TT partition. Fig. 14 shows the overlapping area in the TT and BT partitions. For example, the A section of the BT and TT

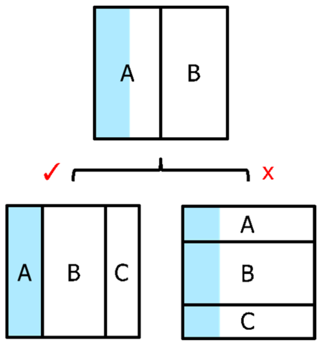


FIGURE 14. Overlapping area in the TT and BT partitions.

vertical partitions includes the pixels of the blue area, but the A section of the TT horizontal partition contains only partial pixels of the blue area. Therefore, the complexity of the TT partition was reduced based on these features in overlapping coverage detection. The cost of the QT, the BT horizontal, and BT vertical partitions were recorded and compared to determine the optimal partition mode. The following four conditions apply to decisions on whether to skip the CU partition, according the partition mode.

1) If the previous best mode is QT partition, the CU skips the TT horizontal and TT vertical partitions.
2) If the previous best mode is BT vertical partition, the CU skips the TT horizontal partition.
3) If the previous best mode is BT horizontal partition, the CU skips the TT vertical partition.
4) If the cost of previous modes is equal, the CU will not skip.

The previous prediction is also included; when the prediction skips the BT and TT partition, this algorithm is not executed.

C. GRADIENT-BASED INTRA MODE DECISION METHOD

The last subsection introduces the texture analysis–based TT and BT partition decision strategy. In this section, the original process of intra prediction is briefly described, and then the proposed method is introduced. The intra prediction of VVC is based on three steps [5], [6]. The first step is Rough Mode Decision (RMD). For the 65 angular modes in VVC, the sum of absolute transform difference (SATD) with Hadamard transform and sum of absolute differences (SAD) are performed to calculate their costs; then, top N with the lowest cost modes is selected from the cost list. Equations (14) and (15) express the cost calculation, C_{SATD} and C_{SAD} are SATD and SAD, respectively, D_{SATD} is the residual of the SATD, D_{SAD} is the residual of the SAD, λ is the Lagrange multiplier, and $Bits_m$ is the number of bits for the prediction mode.

$$C_{SATD} = D_{SATD} + \lambda \times Bits_m \tag{14}$$

$$C_{SAD} = D_{SAD} + \lambda \times Bits_m \tag{15}$$

The second step is the most probable mode (MPM), which merges the modes derived from neighboring CUs into the M

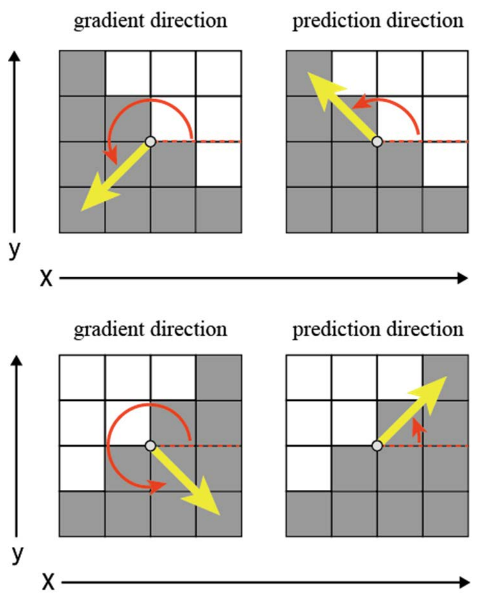


FIGURE 15. Illustrations of correlation between gradient direction and prediction direction.

RMD mode list and establishes the final candidate list. The final step uses the final candidate list to perform RDO [7] and select the optimal intra mode according to the minimal RD-cost. Because the final step is the most time consuming, its complexity is reduced by controlling the number of modes in the final candidate list. The proposed algorithm for intra mode prediction has two parts: texture direction detection and MPM determination.

1) TEXTURE DIRECTION DETECTION

Before intra prediction, the main direction is calculated by using the result of G_x and G_y

$$\theta(i,j) = arctan(\frac{G_y(i,j)}{G_x(i,j)}) \tag{16}$$

$$\phi\,(i,j) = \begin{cases} \theta\,(i,j) + 90^\circ, & \theta\,(i,j) < 90^\circ \\ \theta\,(i,j) - 90^\circ, & \theta\,(i,j) < 270^\circ \\ \theta\,(i,j) - 270^\circ, & \theta\,(i,j) \geq 270^\circ \end{cases} \tag{17}$$

in the block texture detection. The angular $\theta(i,j)$ of gradient direction is calculated by (16) for all pixels in CU. i and j represent the position of the current center pixel in a row and in a column, respectively. After calculating all angular gradient directions in the CU, gradient direction is required to transform gradient direction into a predicted direction because the gradient represents the direction that is perpendicular to the direction of texture. The angular difference between the gradient and predicted direction $\phi(i,j)$ is 90°, as depicted in Fig. 15. The relationship with gradient and predicted direction is calculated according to (17). Our proposed model maps the predicted direction $\phi(i,j)$ into 65 intra modes intra modes and calculates each one's respective gradient G, and then investigates the maximum gradient intra mode ($mode_m$).

The gradient of a modem being too small implies that the current block is homogeneous; thus, we simply add the DC, planar mode, and the modes of the horizontal and vertical directions into our candidate subset, as indicated in (18). If the block is not homogeneous, five modes are added from the left and right side of the $mode_m$, DC mode, and planar mode into the candidate list as shown in (18). Because the correlation with the neighboring block is high, the intra mode of the current CU may be equivalent to the optimal mode of the neighboring CU. The intra modes of the neighboring CU are also added into our method's candidate list. The neighboring CUs are the blocks at the top and to the left of the current block, which are denoted as $mode_t$ and $mode_l$, respectively.

$$cand_l = \begin{cases} \left\{\begin{array}{c} DC\_mode, \\ Planar\_mode, \\ angularmode48, \\ angularmode49, \\ angularmode50, \\ angularmode51, \\ angularmode52, \\ angularmode20, \\ angularmode19, \\ angularmode18, \\ angularmode17, \\ angularmode16, \end{array}\right\}, & if\ flatten \\ \left\{\begin{array}{c} DC\_mode, \\ Planar\_mode, \\ ((mode_m + 58)\,mod65) + 2, \\ ((mode_m + 59)\,mod65) + 2, \\ ((mode_m + 60)\,mod65) + 2, \\ ((mode_m + 61)\,mod65) + 2, \\ ((mode_m + 62)\,mod65) + 2, \\ mode_m, \\ ((mode_m - 1)\,mod65) + 2, \\ (mode_m mod65) + 2, \\ ((mode_m + 1)\,mod65) + 2, \\ ((mode_m + 2)\,mod65) + 2, \\ ((mode_m + 3)\,mod65) + 2, \end{array}\right\}, & else \end{cases} \tag{18}$$

The following five cases were applied to construct the final candidate list:

1) Both $mode_t$ and $mode_l$ are angular modes, and $mode_t$ and $mode_l$ are not equal.
2) Both $mode_t$ and $mode_l$ are angular modes, and $mode_t$ and $mode_l$ are equal.
3) The $mode_t$ is an angular mode, and the $mode_l$ is not an angular mode.
4) The $mode_l$ is an angular mode, and the $mode_t$ is not an angular mode.
5) Both $mode_t$ and $mode_l$ are nonangular modes.

In case 1, $mode_t$ and $mode_l$ along with their neighboring modes are added to the final candidate list. Accordingly, the final candidate list is as follows: $cand_l$, left mode of $mode_t$, $mode_t$, right mode of $mode_t$, left mode of $mode_l$, $mode_l$, and right mode of $mode_l$. In case 2, $mode_t$ and its neighboring modes are added to the final candidate list. The final candidate list is as follows: $cand_l$, left mode of $mode_t$, $mode_t$, and right mode of $mode_t$. In case 3, $mode_t$ and its neighboring modes are added to the final candidate list. Because $mode_l$ is a DC or planar mode, these modes have been in $cand_l$; hence, adding them to the final candidate list is unnecessary. The final candidate list is as follows: $cand_l$, left mode of $mode_t$, $mode_t$, and right mode of $mode_t$. In case 4, model and its neighboring modes are added into the final candidate list. Because $mode_t$ is a DC or planar mode, these modes have been in $cand_l$; hence, adding them to the final candidate list is unnecessary. The final candidate list is as follows: $cand_l$, left mode of $mode_l$, $mode_l$, and right mode of $mode_l$. In case 5, adding $mode_t$ and $mode_l$ to the final candidate list is unnecessary because they are not angular modes.

2) MPM DETERMINATION

Jamali *et al.* [26] proposed a method for ordering candidate modes according to SATD cost. According to the SATD cost, Jamali *et al.* [26] classified candidate modes into the two classes of weak contenders with higher costs and powerful contenders with lower costs. The mode with the lowest SATD is one of the most relevant modes and is highly likely to represent the optimal intra mode for the current CU. Therefore, our proposed model uses this feature to store the SATD cost in relation to the final candidate list and retrieved the top three lowest-cost modes from the cost list, which are denoted as $mode_a$, $mode_b$ and $mode_c$. However, if the cost of $mode_a$, $mode_b$ and $mode_c$ are equal, we can assume that the DC mode or planar mode is more suitable for the current CU. Therefore, if $mode_a$, $mode_b$ and $mode_c$ have the same SATD costs, the angular modes in the candidate list are removed (except DC and planar modes). The flow of the algorithm for fast intra mode prediction is depicted in Fig. 16. The modes that execute the first round of SATD calculations are reduced by at most 19 compared with the original 67 modes. Because the neighboring modes of the original 35 modes are added in the first round of SATD, the second round of SATD calculation is no longer required. A maximum of three modes enter the remaining intra prediction. Thereby, the encoding complexity can be reduced. We implemented the aforementioned two algorithms in the VTM 2.0, VTM 4.0 and VTM 11.0. The experimental results are explained in the next section.

IV. EXPERIMENTAL RESULTS

We evaluated our proposed method by using VVC test software and compared the results with those of several related works to validate the efficiency of the proposed algorithm. First, the time-saving and BDBR performance was calculated for each algorithm. Then, the proposed algorithm was implemented in VTM 2.0, VTM 4.0 and VTM 11.0 to evaluate its overall performance and compare VTM 2.0 and VTM 4.0 with the same version employed in several related works to validate the efficiency of our proposed method, respectively.

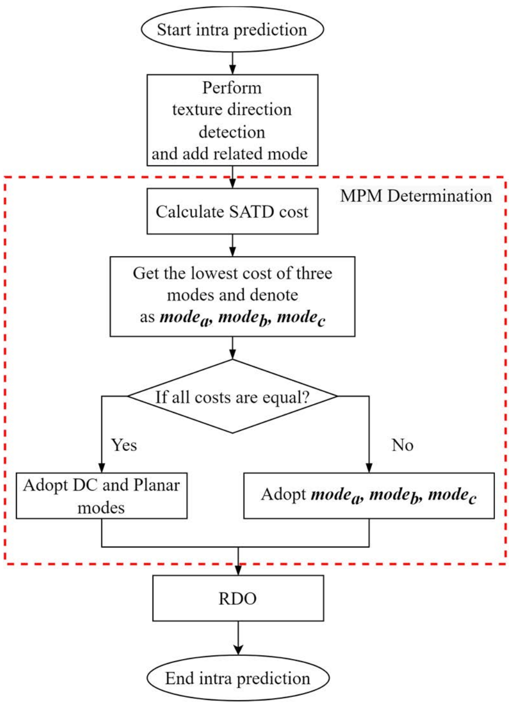

Start intra prediction
Perform texture direction detection and add related mode
MPM Determination
Calculate SATD cost
Get the lowest cost of three modes and denote as $mode_a$, $mode_b$, $mode_c$
If all costs are equal?
Yes
No
Adopt DC and Planar modes
Adopt $mode_a$, $mode_b$, $mode_c$
RDO
End intra prediction

FIGURE 16. Algorithm flow for fast intra mode prediction.

A. EXPERIMENTAL ENVIRONMENT AND CONDITIONS

We used the recently released version of the VVC test software to evaluate the algorithms. All the tests were performed under the all-intra configuration and common test conditions [27]. A total of 28 video sequences from class A1 to class F were used to evaluate our algorithm in terms of BDBR [6] and time saving. Time saving was determined according to (19), where QPS represents the QP set {22, 27, 32, 37}, T_{ori} is the total encoding time of the VTM encoder and T_{mod} is the total encoding time of the VTM encoder with the algorithms. BDBR was determined according to YUV-PSNR and bitrate. The testing machine included an Intel Core i7-8700 CPU 3.20 GHz running Windows 10 (64 bit).

$$TS = \frac{1}{4} \sum_{q \in QPS} \frac{T_{ori}(q) - T_{mod}(q)}{T_{ori}(q)} \times 100 \qquad (19)$$

B. EXPERIMENTAL RESULTS OF THE PROPOSED METHOD

To evaluate the performance of each individual method, we used a part of the test sequences with 100 frames. The results are summarized in Table 1, where TBPSTA is TT and BT partition strategy based on texture analysis, GIMD is Gradient-based intra mode decision method and TS is time saving. Notably, TBPSTA reduced the complexity of the encoding process by approximately 45.30% on average, with a slight BDBR increase. The maximum and minimum time-saving gains were 49.94% and 42.87% on FoodMarket4

TABLE 1. The coding performance of the proposed individual method compared with VTM 4.0.

Class	Test Sequence	TBPSTA (100 Frame)		GIMD (100 Frame)	
		BDBR(%)	TS(%)	BDBR(%)	TS(%)
A1	FoodMarket4	0.28	49.94	0.09	17.04
A2	CatRobot1	0.42	43.82	0.21	22.21
B	Cactus	0.40	42.29	0.15	15.66
	Kimono	0.20	47.72	0.08	20.18
E	FourPeople	0.71	44.72	0.20	16.52
	Johnny	0.58	47.07	0.31	17.34
C	PartyScene	0.31	43.83	0.18	17.03
	RaceHorsesC	0.34	42.87	0.11	16.50
D	BasketballPass	0.64	45.52	0.38	13.28
	Average	0.43	45.30	0.19	17.30

TABLE 2. The coding performance of the proposed overall method compared with anchor method and previous works by VTM 4.0.

Class	Test Sequence	[19]		[20]		[23]		Proposed	
		BDBR(%)	TS(%)	BDBR(%)	TS(%)	BDBR(%)	TS(%)	BDBR(%)	TS(%)
A1	Campfire	-	-	0.78	48.77	0.80	33.00	0.53	52.57
	FoodMarket4	-	-	0.84	52.53	0.64	33.00	0.44	58.89
	Tango2	-	-	0.97	56.95	0.58	29.00	0.61	50.39
A2	CatRobot1	-	-	1.08	54.55	1.28	32.00	0.77	53.23
	DaylightRoad2	-	-	0.78	53.63	1.26	31.00	0.37	47.05
	ParkRunning3	-	-	0.81	52.67	0.69	35.00	0.21	46.07
B	BasketballDrive	3.09	65.26	-	-	1.24	33.00	0.55	47.36
	BQTerrace	1.16	49.44	1.07	55.21	1.00	31.00	0.46	48.03
	Cactus	1.74	52.82	-	-	1.08	34.00	0.55	46.12
	Kimono	1.72	66.59	1.03	55.52	-	-	0.40	55.41
	ParkScene	1.28	56.28	1.33	57.78	-	-	0.45	48.46
	MarketPlace	-	-	-	-	0.59	34.00	0.30	46.15
	RitualDance	-	-	-	-	0.99	34.00	0.65	51.65
E	FourPeople	2.55	62.18	0.85	54.34	1.38	35.00	0.82	49.96
	Johnny	3.07	62.55	0.85	51.46	1.44	33.00	0.76	52.42
	KristenAndSara	2.56	60.82	1.32	52.62	1.19	33.00	0.77	48.40
C	BasketballDrill	1.91	53.05	0.97	59.65	1.67	34.00	0.96	46.37
	BQMall	1.79	56.49	-	-	1.42	35.00	0.69	50.00
	PartyScene	0.28	41.71	0.96	56.65	0.76	36.00	0.42	46.01
	RaceHorsesC	0.84	52.07	0.92	56.71	0.82	36.00	0.47	48.66
D	BasketballPass	2.02	54.16	-	-	1.23	35.00	0.70	50.50
	BlowingBubbles	0.49	43.90	0.69	58.61	0.74	35.00	0.51	45.16
	BQSquare	0.17	32.34	0.73	56.12	0.61	34.00	0.67	49.10
	RaceHorses	0.54	44.93	0.76	54.67	0.95	36.00	0.51	49.83
	Average	1.57	53.41	0.93	54.91	1.02	33.68	0.56	49.49

and Cactus, respectively; the gains in relation to time saving and BDBR were consistent and unaffected by variations in the resolution or content. This suggests that the proposed method can efficiently skip redundant splits in QTMT partition structures. The results also reveal that GIMD can reduce encoding time by approximately 17.30% and negligibly reduce BDBR. This demonstrates that GIMD can efficiently reduce the number of modes required for the RDO process with an average BDBR increase of only 0.19%. Table 1 shows that the proposed strategies are efficient, consistently reduce complexity with various QPs, and possess good generalizability.

TABLE 3. **The coding performance of the proposed overall method compared with anchor method and previous works by VTM 2.0.**

Class	Test Sequence	[4]		[28]		[29]		[24]		Proposed	
		BDBR(%)	TS(%)	BDBR(%)	BDBR(%)	TS(%)	TS(%)	BDBR(%)	TS(%)	BDBR(%)	TS(%)
A1	Campfire	0.95	49.7.	0.31	23.6	1.29	49.1	1.11	58.0	0.70	57.1
	FoodMarket4	0.62	53.5	0.36	21.9	0.84	30.3	1.23	39.9	1.04	63.2
	Tango2	0.99	51.0	0.34	22.8	1.38	48.7	1.33	43.3	0.77	65.2
A2	CatRobot1	1.05	44.8	0.47	23.8	1.99	50.4	1.65	42.4	0.91	55.2
	DaylightRoad2	0.43	60.8	0.54	34.4	2.00	54.1	1.57	46.2	0.34	59.5
	ParkRunning3	0.82	53.7	0.20	21.4	0.80	40.6	0.76	43.5	0.21	48.5
B	BasketballDrive	2.25	64.0	0.44	36.1	1.54	50.1	1.49	46.6	0.63	58.4
	BQTerrace	2.07	56.1	0.31	34.2	1.40	56.0	1.16	48.7	0.38	46.9
	Cactus	1.95	56.7	0.33	21.6	1.73	49.8	1.40	45.9	0.47	40.0
	Kimono	1.90	63.9	0.22	36.3	0.88	49.8	0.85	41.4	0.44	55.6
	ParkScene	1.33	56.6	0.24	30.2	1.25	55.9	0.94	44.1	0.40	44.7
E	FourPeople	2.75	57.6	0.71	29.3	2.71	56.3	1.75	44.7	0.73	47.5
	Johnny	2.51	59.2	0.56	29.2	2.77	55.8	1.67	42.8	0.77	51.8
	KristenAndSara	3.29	59.0	0.46	22.6	2.17	52.8	1.60	43.1	0.67	49.4
C	BasketballDrill	2.01	48.2	0.57	28.4	2.05	54.8	2.30	46.3	1.07	46.3
	BQMall	2.15	55.2	0.66	34.9	2.12	58.9	1.66	46.3	0.59	41.9
	PartyScene	0.60	45.7	0.27	25.0	1.01	51.0	1.14	50.9	0.44	36.3
	RaceHorsesC	1.16	48.4	0.29	23.1	1.35	50.6	1.13	44.6	0.37	45.0
D	BasketballPass	2.33	45.9	0.48	40.3	1.77	53.1	1.49	47.8	0.67	45.5
	BlowingBubbles	0.77	41.6	0.32	26.7	1.40	54.9	1.29	50.8	0.48	42.2
	BQSquare	0.81	46.1	0.27	34.2	0.75	52.8	0.90	46.3	0.73	40.5
	RaceHorses	0.86	43.2	0.30	25.3	1.28	54.7	1.18	45.0	0.49	44.0
F	BasketballDrillText	1.82	47.7	0.49	25.0	2.09	56.3	2.11	51.6	0.98	45.4
	ChinaSpeed	1.30	52.7	0.30	34.7	0.84	48.4	1.21	45.8	0.36	40.0
	SlideEditing	1.12	48.9	0.17	36.6	0.52	45.4	0.95	50.8	0.23	46.3
	SlideShow	2.61	57.4	0.15	28.1	2.11	45.8	1.30	45.5	0.70	46.0
	Average	1.56	52.6	0.38	28.8	1.54	51.0	1.35	46.2	0.59	48.5

The Table 2 and Table 3 provides the results of the proposed algorithm compared with the latest fast methods. We compared the proposed overall scheme with the latest fast methods of H.266/VVC, including those of Chen *et al.* [19], Zhang *et al.* [20] and Park and Kang [23] by using VTM 4.0 and the other latest fast methods of H.266/VVC, including those of Yang *et al.* [4], Fu *et al.* [28], Chen *et al.* [29] and Li *et al.* [24] by using VTM 2.0. These methods were efficient, fast, and typical of VVC encoder methods. The methods of Yang *et al.* [4], Chen *et al.* [19], Zhang *et al.* [20], Park and Kang [23], Fu *et al.* [28], Chen *et al.* [29] and Li *et al.* [24] were compared by using the same version of the VVC test software. The Table 2 reveals that our proposed method can reduce encoding time by approximately 49.49%, with a BDBR increase of only 0.56%. Our method's BDBR increase was almost negligible, and its encoding time was half that of the VTM 4.0. The increment of the bitrate and the decrement of the YUV Peak signal-to-noise ratio (YUV-PSNR) between the VTM 4.0 and our proposed method are only 0.15% and 0.03%, respectively. Among the five methods, the BDBR of the proposed method was the lowest because the accuracy of split ratio was 94%. This means that the proposed method could ensure the greatest number of CUs (that required splitting) were not wrongly skipped, and this is reflected in the BDBR. The time savings registered by the algorithms of Chen *et al.* [19], Zhang *et al.* [20], as well as our proposed method were similar, but the BDBR of the proposed method was lower than that of the algorithms of Chen *et al.* [19] and Zhang *et al.* [20]. Table 2 indicates that the average BDBR of the proposed method, [19], [20] and [23] increased by 0.56%, 1.57%, 0.93% and 1.02%, respectively. Compared with the proposed method, the average BDBR of [19], [20] and [23] methods were greater by 1.01%, 0.37% and 0.46%, respectively. Furthermore, the time saving of the proposed overall method was superior to that of [23] methods and close to that of [19] and [20]. Table 2 shows that the time saving of the [4] and [23] methods was greater than that of the proposed method by 12.75%, 16.07%, respectively. To further investigate the coding performance of the proposed approach on different VTM, the proposed method is also implemented to VTM 11.0 and evaluated it by test sequences. Table 3 indicates the proposed method achieves 48.5% encoding time saving on average with 0.59% BDBR increase on average.

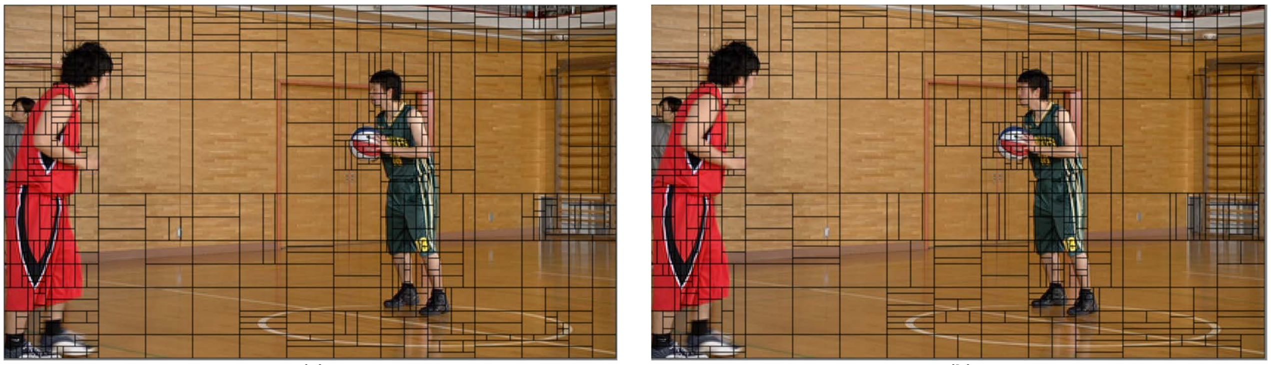


FIGURE 17. CU partition results of various algorithms. (a) Partition of BasketballPass by using VTM 11.0. (b) Partition of BasketballPass by using our method.

TABLE 4. The coding performance of the proposed overall method in VTM 11.0.

Class	Test Sequence	Proposed	
		BDBR(%)	TS(%)
3840 x 2160 Class A1	Campfire	0.26	43.58
	FoodMarket4	0.52	50.29
	Tango2	0.58	47.35
3840 x 2160 Class A2	CatRobot1	0.37	41.43
	DaylightRoad2	0.26	33.29
	ParkRunning3	0.27	41.57
1920 x 1080 Class B	BasketballDrive	0.36	44.73
	BQTerrace	0.30	40.05
	Cactus	0.40	42.00
	Kimono	0.34	43.03
	ParkScene	0.37	40.00
	MarketPlace	0.58	40.41
	RitualDance	0.43	42.55
1280 x 720 Class E	FourPeople	0.62	39.57
	Johnny	0.62	44.16
	KristenAndSara	0.55	41.19
832 x 480 Class C	BasketballDrill	1.16	47.88
	BQMall	0.52	45.43
	PartyScene	0.32	41.50
	RaceHorsesC	0.29	44.90
416 x 240 Class D	BasketballPass	0.44	42.02
	BlowingBubbles	0.39	40.66
	BQSquare	0.36	40.30
	RaceHorses	0.37	41.54
	Average	0.44	42.47

Table 3 shows the average BDBR values of Yang *et al.* [4], Fu *et al.* [28], Chen *et al.* [29] and Li *et al.* [24] increased by 1.56%, 0.38%, 1.54% and 1.36%, respectively. Meanwhile, their time saving are 52.6%, 28.8%, 51.0% and 46.2% for all video sequences, respectively. Compared with Yang *et al.* [4], Chen *et al.* [29] and Li *et al.* [24], our proposed method achieves similar time saving on average but the BDBR of our method was lower than these works. Table 4 lists the results of this VTM 11.0 implementation; the average BDBR of the proposed method was 0.44%, and the average time saving of the proposed method was 42.47%. Because VTM 11.0 has added some techniques for skipping redundant partitions and intra mode prediction, the time saving capability in VTM 11.0 is less than that in VTM 4.0 and VTM 2.0. Table 2, Table 3 and Table 4 demonstrate that the proposed method is efficient and offers consistent complexity reduction with various test sequences as well as good generalizability. Fig. 17 illustrates the splitting result of the default algorithm in VTM 11.0 and the proposed method with the QP set to 37. The CU partition is skipped if the block is flat, and the split is close to the textures if the block is complex.

V. CONCLUSION

In this paper, a texture analysis–based TT and BT partition strategy and Gradient-based Intra Made Decision method are proposed to reduce the redundant and time-consuming processing in CU partitioning and intra prediction, respectively. These techniques are based on texture features, and we used these features to build the regression function to reduce the redundant processes both in CU partition and intra prediction efficiently. Compared with fixed threshold, the proposed methods can be more flexible for various texture and various QP. The experimental results revealed that the proposed method can save 49.49%, 48.5% in encoding time and only increased the BDBR by 0.56%, 0.59% compared with the VTM 4.0 and VTM 2.0, respectively. Additionally, in VTM 11.0, the time saving of the proposed method was 42.47% with a BDBR increase of 0.44%. Regardless of the VTM version used, the proposed method delivered favorable results and outperformed several well-known fast methods. In summary, our proposed method achieves high efficiency encoding, which implies that it achieves more consistent encoding time saving for all test sequences with slight BDBR increased in various version.

ACKNOWLEDGMENT

The authors would like to thank Dr. Songan Mao from Qualcomm, Ching-Cheih Lin, a Researcher from Industrial

Technology Research Institute, and anonymous reviewers and editors for their valuable comments and suggestions to improve the quality of the paper.

REFERENCES

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

[2] B. Bross, *Versatile Video Coding (Draft 1)*, document JVET-J1001, Joint Video Exploration Team (JVET), Geneva, Switzerland, Apr. 2018.

[3] B. Bross, Y.-K. Wang, Y. Ye, S. Liu, J. Chen, G. J. Sullivan, and J.-R. Ohm, "Overview of the versatile video coding (VVC) standard and its applications," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 31, no. 10, pp. 3736–3764, Oct. 2021.

[4] H. Yang, L. Shen, X. Dong, Q. Ding, P. An, and G. Jiang, "Low-complexity CTU partition structure decision and fast intra mode decision for versatile video coding," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 30, no. 6, pp. 1668–1682, Jun. 2020.

[5] Y. Piao, *Encoder Improvement of Unified Intra Prediction*, document JCTVC-C207, Jan. 2013.

[6] J. Chen, Y. Chen, M. Karczewicz, X. Li, H. Liu, L. Zhang, and X. Zhao, "Coding tools investigation for next generation video coding based on HEVC," *Proc. SPIE*, vol. 9599, Sep. 2015, Art. no. 95991B.

[7] G. J. Sullivan and T. Wiegand, "Rate-distortion optimization for video compression," *IEEE Signal Process. Mag.*, vol. 15, no. 6, pp. 74–90, Nov. 1998.

[8] G. Bjontegaard, *Calculation of Average PSNR Differences Between RD-Curves*, document VCEG-M33, 2001.

[9] Z. Liu, X. Yu, Y. Gao, S. Chen, X. Ji, and D. Wang, "CU partition mode decision for HEVC hardwired intra encoder using convolution neural network," *IEEE Trans. Image Process.*, vol. 25, no. 11, pp. 5088–5103, Nov. 2016.

[10] F. Duanmu, Z. Ma, and Y. Wang, "Fast CU partition decision using machine learning for screen content compression," in *Proc. IEEE Int. Conf. Image Process. (ICIP)*, Sep. 2015, pp. 4972–4976.

[11] Y. Zhang, S. Kwong, X. Wang, H. Yuan, Z. Pan, and L. Xu, "Machine learning-based coding unit depth decisions for flexible complexity allocation in High Efficiency Video Coding," *IEEE Trans. Image Process.*, vol. 24, no. 7, pp. 2225–2238, Jul. 2015.

[12] L. Zhu, Y. Zhang, Z. Pan, R. Wang, S. Kwong, and Z. Peng, "Binary and multi-class learning based low complexity optimization for HEVC encoding," *IEEE Trans. Broadcasting*, vol. 63, no. 3, pp. 547–561, Sep. 2017.

[13] X. Liu, Y. Li, D. Liu, P. Wang, and L. T. Yang, "An adaptive CU size decision algorithm for HEVC intra prediction based on complexity classification using machine learning," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 29, no. 1, pp. 144–155, Jan. 2019.

[14] Z. Jin, P. An, C. Yang, and L. Shen, "Fast QTBT partition algorithm for intra frame coding through convolutional neural network," *IEEE Access*, vol. 6, pp. 54660–54673, 2018.

[15] Z. Jin, P. An, L. Shen, and C. Yang, "CNN oriented fast QTBT partition algorithm for JVET intra coding," in *Proc. IEEE Vis. Commun. Image Process. (VCIP)*, Dec. 2017, pp. 1–4.

[16] G. Tang, M. Jing, X. Zeng, and Y. Fan, "Adaptive CU split decision with pooling-variable CNN for VVC intra encoding," in *Proc. IEEE Vis. Commun. Image Process. (VCIP)*, Dec. 2019, pp. 1–4.

[17] Z. Wang, S. Wang, J. Zhang, S. Wang, and S. Ma, "Effective quadtree plus binary tree block partition decision for future video coding," in *Proc. Data Compress. Conf. (DCC)*, Apr. 2017, pp. 23–32.

[18] T. Amestoy, A. Mercat, W. Hamidouche, D. Menard, and C. Bergeron, "Tunable VVC frame partitioning based on lightweight machine learning," *IEEE Trans. Image Process.*, vol. 29, pp. 1313–1328, 2019.

[19] J. Chen, H. Sun, J. Katto, X. Zeng, and Y. Fan, "Fast QTMT partition decision algorithm in VVC intra coding based on variance and gradient," in *Proc. IEEE Vis. Commun. Image Process. (VCIP)*, Dec. 2019, pp. 1–4.

[20] Q. Zhang, Y. Wang, L. Huang, and B. Jiang, "Fast CU partition and intra mode decision method for H. 266/VVC," *IEEE Access*, vol. 8, pp. 117539–117550, 2020.

[21] Q. Zhang, Y. Zhao, B. Jiang, L. Huang, and T. Wei, "Fast CU partition decision method based on texture characteristics for H. 266/VVC," *IEEE Access*, vol. 8, pp. 203516–203524, 2020.

[22] Y. Fan, H. Sun, J. Katto, and J. Ming, "A fast QTMT partition decision strategy for VVC intra prediction," *IEEE Access*, vol. 8, pp. 107900–107911, 2020.

[23] S.-H. Park and J.-W. Kang, "Context-based ternary tree decision method in versatile video coding for fast intra coding," *IEEE Access*, vol. 7, pp. 172597–172605, 2019.

[24] Y. Li, G. Yang, Y. Song, H. Zhang, X. Ding, and D. Zhang, "Early intra CU size decision for versatile video coding based on a tunable decision model," *IEEE Trans. Broadcast.*, vol. 67, no. 3, pp. 710–720, Sep. 2021.

[25] *Coding of Moving Video: High Efficiency Video Coding*, document ITU-T, Apr. 2015.

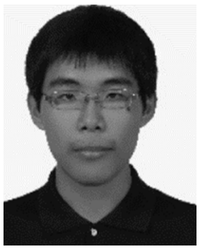
[26] M. Jamali, S. Coulombe, and F. Caron, "Fast HEVC intra mode decision based on edge detection and SATD costs classification," in *Proc. Data Compress. Conf.*, Apr. 2015, pp. 43–52.

[27] F. Bossen, J. Boyce, K. Suehring, X. Li, and V. Seregin, *VTM Common Test Conditions and Software Reference Configurations for SDR Video*, document JVET-t2010, Teleconference, Oct. 2020.

[28] T. Fu, H. Zhang, F. Mu, and H. Chen, "Fast CU partitioning algorithm for H.266/VVC intra-frame coding," in *Proc. IEEE Int. Conf. Multimedia Expo. (ICME)*, Jul. 2019, pp. 55–60.

[29] F. Chen, Y. Ren, Z. Peng, G. Jiang, and X. Cui, "A fast CU size decision algorithm for vvc intra prediction based on support vector machine," *Multimedia Tools Appl.*, vol. 79, no. 37, pp. 27923–27939, 2020.

CHI-TING NI received the B.S. degree in engineering science from the National Cheng Kung University, Tainan, Taiwan, in 2019, where she is currently pursuing the Ph.D. degree in computer science and information engineering. Her current research interests include image processing, very large-scale integrated chip design, and embedded systems.



SHIH-HSIANG LIN received the B.S. and Ph.D. degrees in computer science and information engineering from the National Cheng Kung University, Tainan, Taiwan, in 2014 and 2018, respectively. His current research interests include image processing, very large-scale integrated chip design, and embedded systems.

PEI-YIN CHEN (Senior Member, IEEE) received the B.S. degree in electrical engineering from the National Cheng Kung University, Tainan, Taiwan, in 1986, the M.S. degree in electrical engineering from The Pennsylvania State University, University Park, PA, USA, in 1990, and the Ph.D. degree in electrical engineering from the National Cheng Kung University, in 1999. He is currently a Professor with the Department of Computer Science and Information Engineering, National Cheng Kung University. His research interests include very large scale integration chip design, video compression, fuzzy logic control, and gray prediction.

YU-TING CHU received the B.S. degree in engineering science from the National Sun Yat-sen University, Kaohsiung, Taiwan, in 2018, and the M.S. degree in computer science and information engineering from the National Cheng Kung University, Tainan, Taiwan, in 2020. Her current research interests include image processing, very large-scale integrated chip design, and embedded systems.

• • •