symmetry

MDPI

Article

A Fast CU Partition Algorithm Based on Gradient Structural Similarity and Texture Features

Zhiyong Jing, Peng Li, Jinchao Zhao and Qiuwen Zhang *

College of Computer and Communication Engineering, Zhengzhou University of Light Industry, Zhengzhou 450002, China
* Correspondence: 2012032@zzuli.edu.cn; Tel.: +86-371-8660-9559

Abstract: The H.266/Versatile Video Coding (VVC) standard poses a great challenge for encoder design due to its high computational complexity and long encoding time. In this paper, the fast partitioning decision of coding blocks is investigated to reduce the computational complexity and save the coding time of VVC intra-frame predictive coding. A fast partitioning algorithm of VVC intra-frame coding blocks based on gradient structure similarity and directional features is proposed. First, the average gradient structure similarity of four sub-coding blocks under the current coding block is calculated, and two thresholds are set to determine whether the current coding block terminates the partitioning early or performs quadtree partitioning. Then, for the coding blocks that do not satisfy the above thresholds, the standard deviation of the vertical and horizontal directions of the current coding block is calculated to determine the texture direction and skip unnecessary partitioning to reduce computational complexity. Based on the VTM10.0 platform, this paper evaluates the performance of the designed fast algorithm for partitioning within the VVC coding unit. Compared with VTM10.0, the encoding rate is improved by 1.38% on average, and the encoder execution time is reduced by 49.32%. The overall algorithm achieves a better optimization of the existing VVC intra-frame coding technique.

Keywords: VVC; fast split; GDSIM; texture features

Citation: Jing, Z.; Li, P.; Zhao, J.; Zhang, Q. A Fast CU Partition Algorithm Based on Gradient Structural Similarity and Texture Features. *Symmetry* **2022**, *14*, 2644. https://doi.org/10.3390/sym14122644

Academic Editor: Nicola Mastronardi

Received: 10 November 2022
Accepted: 6 December 2022
Published: 14 December 2022

1. Introduction

With the advent of the information age, people have higher requirements for video resolution in daily life, which brings great pressure to the transmission of video on the network. On the one hand, the demand for ultra-high definition (UHD) video is increasing day by day, and the data volume of video data is also increasing with the improvement of clarity. On the other hand, due to the high-speed circulation of information, the video flow on the communication network also grows rapidly. Video coding is not only the key technology for efficient storage and transmission of multimedia information but also an important part of modern information technology. At present, one of the development trends of video technology is to seek a higher resolution and better clarity, to express natural scenes more realistically and clearly. For example, Japan Broadcasting Corporation (NHK) has been committed to the compression and transmission technology research of 4K (3840 × 2160) and even 8K (7680 × 4320) UHD video programs [1]. However, ultra-high resolution leads to a sharp increase in video data, which makes it very difficult to store and transmit video data [2]. Currently, the relatively perfect international video coding standard high-efficiency video coding (HEVC) is mainly oriented to HD (720P, 1080I, 1080P) video coding [3], which is unable to meet the current requirements of emerging video coding such as ULTRA-high definition, high dynamic range, and 360° VR. Therefore, the research and standard formulation of the next generation ultra-high definition video coding technology has become very urgent. Similar to the advent of H.265/HEVC, H.266/VVC is for further optimizing the compression, which can save about 50% of the data traffic and at

the same time ensure that the definition of video transmission remains unchanged, so VVC has obvious advantages.

Many new intra-frame coding techniques have been proposed in VVC, such as the quadtree nested multi-type tree (QTMT) partition method, 67 intra-prediction modes, Position Dependent Intra Prediction Combination (PDPC), Linear model intra-prediction techniques across luma-chroma, Multi-reference line Intra prediction technology, Intra-frame prediction sub-block partitioning techniques, Matrix-weighted average intra-frame prediction techniques, etc. [4,5]. Among them, the newly introduced multi-type tree (MTT) partition process occupies a large amount of intra-frame coding time. Disabling MTT can save about 90% of the encoding time, and there is a huge space for optimization [6]. In addition, new key technologies in each module bring high computational complexity to video coding, so more and more researchers began to optimize different modules to improve video coding speed, laying a solid foundation for the latest multi-function video coding standard H.266/VVC.

In the coding structure of HEVC, the coding tree unit (CTU) plays an important role. The CTU is the root node of the quadtree partition in HEVC. At the coding unit layer, each coding unit block will be divided into prediction units (PUs) and transform units (TUs) with symmetrical or asymmetrical structures. As with CU partitioning, for prediction and transformation units PU and TU, the size of PU and TU under each CU will not exceed the size of the current CU. To better improve the coding efficiency, it is necessary to determine the optimal CU partition mode through rate-distortion optimization in the tree coding unit layer (CTU). A key feature of HEVC is that there are multiple division concepts: CU, PU, and TU. However, these concepts do not exist in VVC, and CU is used as a uniform representation.

The new QTMT partition method in VVC makes CU partition more flexible, adding rectangular blocks no longer limited to squares. VVC still adopts the CTU structure in HEVC. In particular, the CTU will first perform quad-tree partition (QT), and then further conduct multi-type tree partition. Multi-type tree partition includes four partition methods, as shown in Figure 1, which are vertical binary tree division (BTV), horizontal binary tree division (BTH), vertical ternary tree division (TTV), and horizontal ternary tree division (TTH). In VVC, the maximum size allowed for the root node of the quadtree is 128 × 128 while the minimum size allowed for the child node of the quadtree is 16 × 16. But the maximum size allowed for the root node of the ternary tree is 64 × 64 and the maximum size allowed for the child node of the ternary tree is 4 × 4. Moreover, the maximum size allowed for the root node of the binary tree is 64 × 64 and the minimum size allowed for the child node of the binary tree is 4 × 4. For a clearer understanding of the coding unit partition process, Figure 2 shows an example diagram.

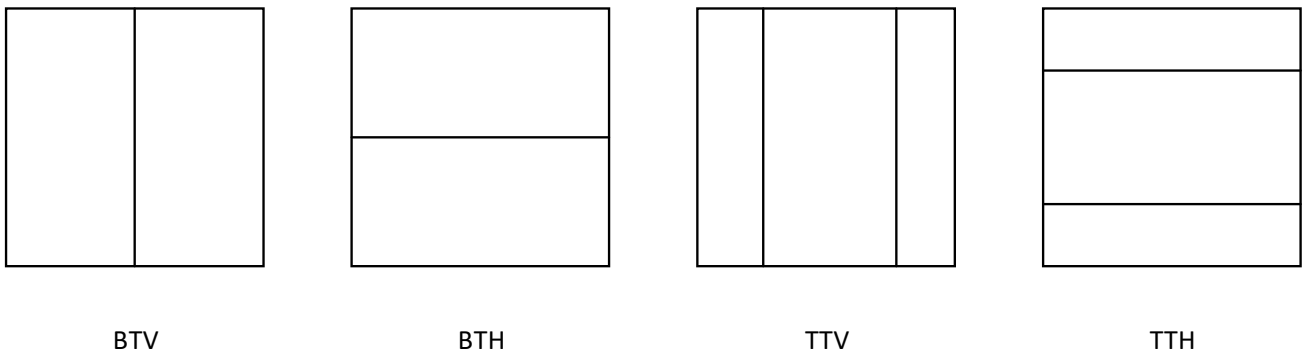

Figure 1. Multi-type tree division.

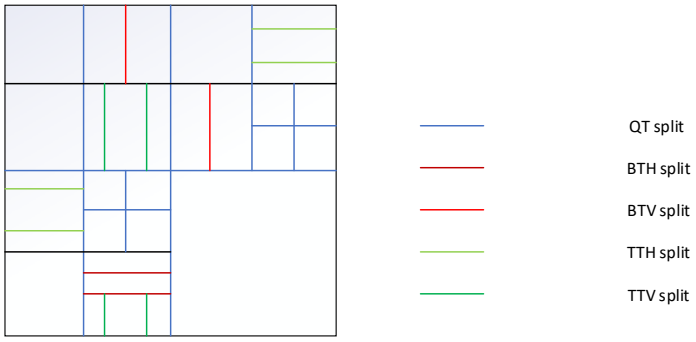

Figure 2. CTU division diagram.

In the CU partition process of VVC, five partition modes need to be traversed at most when using VTM for coding. Under the same depth, CUs are divided in the order of QT, BTH, BTV, TTH, and TTV. To obtain the CU optimal partition decision, all possible partition structure combinations are iteratively tried to select the partition structure with the lowest distortion cost. Although this partitioning method can obtain the globally optimal partition structure, it also consumes a lot of time. After adding binary tree (BT) and ternary tree (TT) to VVC division, the order of selecting division modes may be inconsistent during the traversal process, but the obtained division results are the same. To this end, we can first carry out a horizontal binary tree partition, and again carry out a horizontal binary tree partition for the two sub-blocks respectively, to obtain four sub-blocks in the same horizontal direction. The result is the same as that obtained by first performing one horizontal ternary tree partition and then dividing the middle block by a horizontal binary tree.

The way and order of CU partitioning are different, and the same partition structure may be obtained, this situation is called partition redundancy. Partitioning redundancy increases the computational complexity of coding by repeatedly computing the rate-distortion generation values of the same partitioning structure. To avoid similar situations, VVC imposes some restrictions on the multi-fork tree partitioning process. In the above case, VVC prohibits the middle part of the ternary tree from performing the partition with the binary tree in the same direction as last time. In addition, partition restrictions also occur in the following cases. As shown in Figure 3, after the square block is divided into a vertical binary tree, the horizontal binary tree division is performed on both the left and right sub-blocks. The result will be consistent with the direct quad-tree division of the parent block, and four identical square sub-blocks are obtained. Since the right sub-block is partitioned first, the horizontal binary tree splitting of the left sub-block should be prohibited. And if the parent block first performs horizontal binary tree splitting, and the sub-block does vertical binary tree splitting, the same result will be obtained. Therefore, the lower sub-block should be partitioned, and the vertical binary tree splitting of the upper sub-block should be prohibited.

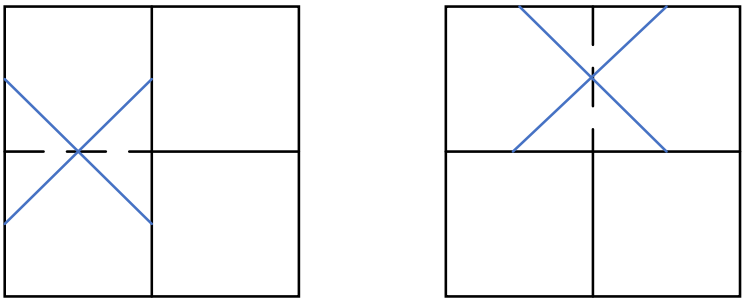

Figure 3. Limitations of VVC on the partitioning process.

The new multi-type tree partition methods in VVC bring a heavy amount of calculation, followed by a high computational burden of intra-frame coding. The standard encoder traverses all partitioning possibilities to find the optimal partitioning method when making multi-type tree partitioning decisions. Figure 4 shows the result of the reduction of the coding time by the binary tree and ternary tree partition [6]. From this, it can be seen that the structure of the quadtree nested multi-type tree added by VVC affects the coding time. When the VTM encoder forbids binary tree partitioning, the average coding time of each video sequence is saved by 75%, while when the coding blocks are prohibited from ternary tree partition, the average coding time of each video sequence is saved by 48%. Furthermore, when both BT and TT splits are disabled, the average encoding time is reduced by about 92%. Analysis shows that multi-type tree partitioning takes up a lot of coding time. Therefore, research that focuses on reducing BT/TT partitioning can achieve significant coding complexity reduction, which is more conducive to the real-time application of encoders.

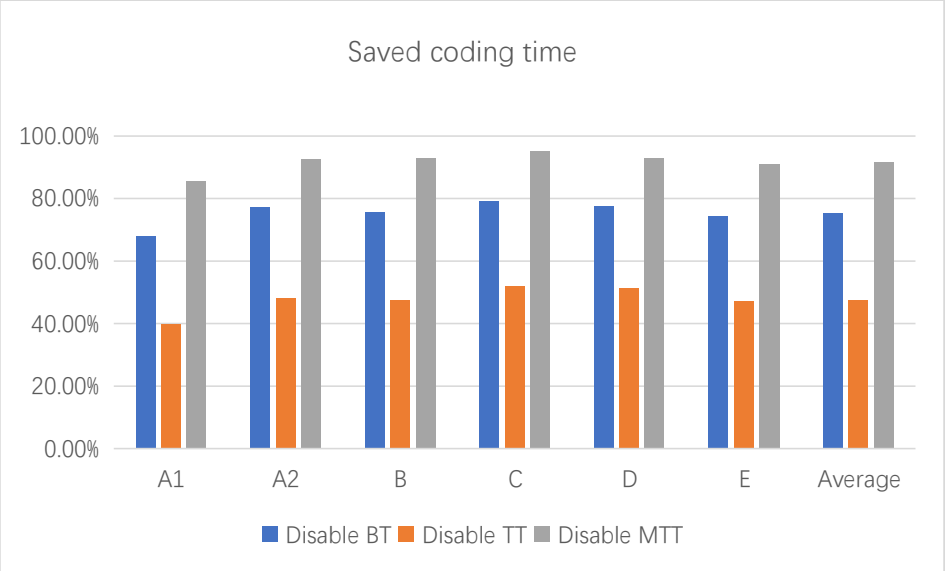

Figure 4. Encoding time is saved under different conditions in reference [6].

Our main contributions in this paper are: A fast CU partition in the algorithm for VVC is proposed. In the quadtree partition stage, the image edge information can be extracted well according to the similarity of gradient structure, and the average gradient structure similarity of coding units can be predicted in the early stage of intra-frame prediction to terminate the partition in advance or carry out the quadtree partition. In the multi-type tree partition stage, the texture direction of CU is determined according to the standard deviation of different directions, and unnecessary intra-frame partition methods are skipped to reduce the complexity of intra-frame coding. The experimental results show that the proposed algorithm focuses on the rapid partition of coding units. By terminating the division of coding blocks in advance, the process of rate-distortion optimization is reduced, which improves the coding efficiency and saves coding time.

The remaining of this paper is organized as follows. Section 2 reviews the related work for HEVC and VVC. Section 3 presents the fast partitioning algorithm for VVC intra-frame coding block. Section 4 shows the experimental results to verify the effectiveness of our proposed approach. Section 5 concludes the paper.

2. Related Works

With the continuous development of video coding standards, the coding performance is also improving, but it also causes the explosive growth of visual data and greater coding time complexity. The CU partitioning process for intra-frame coding occupies most of the

coding time. For this reason, most intra-coding fast algorithms focus on how to reduce the complexity of CU partitioning. A large number of scholars have contributed to the complexity reduction of coding time while ensuring coding performance. And their main work is to optimize the relevant algorithms to improve the coding speed.

2.1. Research Status of HEVC Fast Algorithms

Many works have been done to reduce the complexity of CU partition complexity in HEVC. To realize the fast partitioning algorithm of intra-frame coding, a lot of research has been carried out by the predecessors. Mu et al. [7] proposed a fast partition algorithm based on the original pixel gradient texture analysis, which judged the partition method by comparing the gradient texture of the CU, to terminate unnecessary partitions and reduce the coding complexity. Lee et al. [8] utilized grayscale histograms to reduce the depth range of the CU and then compared the sum of the absolute differences of the current depth CU with the corresponding depth rate-distortion (RD) threshold within the depth range, to determine whether to terminate the CU partition. Sun et al. [9] proposed to calculate the Haar wavelet coefficients of the current CU according to the texture features of the CU and compared it with the threshold, thereby saving the coding time of the CU. Du et al. [10] used a random forest classifier for offline learning to speed up the CU partitioning process based on the classification results. Zhang et al. [11] used two features of entropy and texture contrast from the perspective of micro-texture and macro-texture of coding blocks, so that block partitioning can be accurately predicted, which in turn saves time. Zhang et al. [12] proposed a CU partitioning algorithm based on spatial correlation, which predicts the minimum and maximum values of the current CU depth range by the depth of adjacent CUs, thus reducing the number of traversals for quadtree partitioning and saving coding time. Jimenez et al. [13] proposed a CU depth decision algorithm, which determines the CU depth by performing statistical analysis of the CU depth using hypothesis testing. Sun et al. [14] proposed a fast CU partitioning algorithm for HEVC intraframe coding, which first obtains the CU texture complexity from a specific coding block and then uses the texture complexity of the CU to decide whether a CU should continue to be partitioned into four sub-CUs. In addition, fast CU partition algorithms based on machine learning are becoming more and more popular. Xu et al. [15] proposed a fast CU partition decision algorithm based on convolutional neural network (CNN), and designed a hierarchical CNN with early termination (ETH-CNN) to learn to predict CU partition maps. Fan et al. [16] used two convolutional neural networks to decide the partition mode of CU and prediction unit respectively and used the pixel points and quantization parameter (*QP*) information of the encoded image in training the convolutional neural network. The above algorithms are proposed based on the quadtree division structure in HEVC, which optimize the coding process without affecting the coding quality of the encoder, to terminate the CU partition process and rate distortion optimization (RDO) operation in advance. However, they do not apply to the multi-type tree structure proposed in the new standard.

2.2. Research Status of VVC Fast Algorithms

The above algorithms are all oriented to the HEVC coding standard using QT partition, while VVC exploits a more complex quad, ternary, and binary tree coding structure to further compress the video. Combined with the newly added intra-frame coding technology, we find a coding optimization method suitable for VVC. To reduce the coding complexity in VVC, researchers have put forward a large number of fast partitioning algorithms, which can be roughly divided into three categories: the traditional method, the second method based on machine learning, and the third method based on neural networks.

In the traditional method, the CU partition is terminated in advance by acquiring CU characteristics such as CU texture and CU partition depth to save coding time. Zhang et al. [17] proposed a fast intra-frame coding algorithm based on texture features. Among them, the algorithm uses texture energy and texture direction for CU delineation

in advance, thus terminating unnecessary delineation and reducing the coding complexity. Lei et al. [18] calculated and analyzed the distribution of Hadamard cost and rate-distortion cost, and designed a CU partition early termination strategy based on the Bayesian decision criterion. Saldanha et al. [19] judged the texture direction according to the variance and intra-prediction mode of the current CU and skipped unnecessary partition modes according to the texture direction. Cui et al. [20] determined the probability of each partition mode according to the gradients in the horizontal and vertical directions and skipped the division mode with a lower probability.

The second category is machine learning-based methods using decision trees (DT), support vector machines (SVM), and other methods to speed up CU division. Among them, SVM is very popular to accelerate CU partition. Zhao et al. [21] proposed a fast CU division decision algorithm based on SVM, analyzed the ratio of various CU size division patterns, and trained SVM models to accelerate CU division ratio and edge point ratio. Zhang et al. [22] proposed an intra-coding complexity reduction technique based on an improved SVM to classify the CU partition decision. Chen et al. [23] proposed an SVM-based CU partition early termination algorithm, which selects CU entropy and texture contrast to represent the characteristics of CU partition direction, and predicts specific CU partition direction through SVM. Besides support vector machines, there are other machine learning-based algorithms. Huang et al. [24] used the Resnet neural network to predict the division depth range of CU to terminate the CU partition in advance. For the CU that does not meet the termination partition, the classification type is judged according to the random forest classifier, to save the coding time. Yang et al. [25] designed a decision tree-based CU partition mode selection algorithm (CSD-SL), but it needed to train different decision tree models for different quantitative parameters and adopted more characteristic parameters with high complexity. Zhang et al. [26] proposed an algorithm for fast partitioning of coding blocks by using a random forest classifier model and fast intra-frame mode selection based on texture features of video images.

The third category is neural network-based methods. In recent years, deep learning frameworks have developed rapidly, and some researchers exploit trained neural networks to predict CU partition patterns. Tang et al. [27] constructed a CNN model with variable pooling layers, which can flexibly adapt to various CU shapes and determine in advance whether the CU needs to be divided. Tissier et al. [28] took a 64×64 size CU as the input of the CNN network, and calculated the edge probability on the 4×4 block boundary to determine the partition probability, thereby terminating the CU division process early and saving a lot of coding time. Li et al. [29] designed a CNN model that can terminate the division early in each stage, which determined the coding block division process according to the multi-stage flexible QTMT structure, and designed an adaptive loss function for training the CNN model. Park et al. [30] proposed a lightweight neural network (LNN) model that prematurely terminates ternary tree partitioning according to two valid features (EVF and DVF) related to ternary tree partitioning. Li et al. [31] proposed a hierarchical grid-based CNN model, which can directly obtain the full division of parent CU and child CU, reducing the coding time.

Compared with HEVC, the coding structure of VVC is more flexible, and its complexity is several times higher than that of HEVC. The increase in complexity is mainly the consequence of rate-distortion. Therefore, rate-distortion optimization and fast CU partition are the focus of the research. The above-mentioned literature reduces the rate-distortion optimization process through various designed algorithms, and achieves the effect of saving encoding time, but the intra-frame information has not been fully utilized, and there is still room for improving encoding efficiency.

3. Method

In recent years, people have made new breakthroughs in the field of image research, Wang et al. [32,33] proposed a new feature for evaluating image quality: Structural

Similarity (*SSIM*). It consists of three parts, namely, brightness, contrast, and structure, which can be expressed as:

$$SSIM(i,j) = Lum(i,j)Con(i,j)Str(i,j) \tag{1}$$

$$Lum(i,j) = \frac{2\alpha_i\alpha_j + c_1}{\alpha_i^2 + \alpha_j^2 + c_1} \tag{2}$$

$$Con(i,j) = \frac{2\beta_i\beta_j + c_2}{\beta_i^2 + \beta_j^2 + c_2} \tag{3}$$

$$Str(i,j) = \frac{\beta_{ij} + c_3}{\beta_i\beta_j + c_3} \tag{4}$$

where i and j are two image blocks (the proposed algorithm needs to consider the similarity of CU, which will be referred to as CU in the following text), $Lum(i,\ j)$, $Con(i,\ j)$, and $Str(i,j)$ denote the brightness, contrast, structure, respectively. α_i and α_j represent the mean values of the two CUs, β_i and β_j are the standard deviation values of the two CUs, and β_{ij} is the covariance value of the two CUs. c_1, c_2, and c_3 are set constants. To avoid formula denominator of 0, c_1, c_2, and c_3 are set as constants. The higher the SSIM of the two CUs, the more similar the two CUs are.

Although the structural similarity model of the two CUs is relatively simple and has certain advantages compared to the peak signal-to-noise ratio (PSNR) or mean square error (MSE) model, there are also some shortcomings. For severely blurred and distorted images, their texture properties are seriously damaged, and the structural aspect $Str(i,j)$ in *SSIM* does not reflect this change well. Since the gradient is sensitive to the texture features and small detail contrast of the image, it can be a good evaluation of image clarity. We further propose to determine the 32 × 32 or 16 × 16 size partition based on the similarity of the gradient structure.

3.1. The Proposed Fast Cu Partitioning Algorithm

Compared with the Canny operator and the prewitt operator, the calculation of the Sobel operator algorithm is relatively simple and the speed is relatively fast. To simplify the calculation, the gradient calculation of the current CU adopts the simplest Sobel operator in the 3 × 3 template gradient filter, including the vertical edge operator V and the horizontal edge operator H, as shown in Figure 5:

−1	0	+1
−2	0	+2
−1	0	+1

Vertical edge operator V

+1	+2	+1
0	0	0
−1	−2	−1

Horizontal edge operator H

Figure 5. Sobel operator.

The Sobel operator consists of two groups of 3 × 3 matrices, which are in the vertical and horizontal positions, respectively, and are convolved with the image plane to obtain the approximation of the vertical and horizontal brightness differences respectively.

$$Gx = \begin{bmatrix} -1 & 0 & +1 \\ -2 & 0 & +2 \\ -1 & 0 & +1 \end{bmatrix} * A \tag{5}$$

$$Gy = \begin{bmatrix} +1 & +2 & +1 \\ 0 & 0 & 0 \\ -1 & -2 & -1 \end{bmatrix} * A \tag{6}$$

$$G(i,j) = \sqrt{Gx^2 + Gy^2} \tag{7}$$

Among them, A is the original image, Gx is the vertical gradient value of the CU along the $(i,\ j)$ position, Gy is the lateral gradient value of the CU along the $(i,\ j)$ position, $G(i,\ j)$ is the CU at $(i,\ j)$ position of the gradient magnitude.

Liu et al. [34] proposed a new feature to measure the changes in image contrast and structure: gradient similarity (*GSIM*). The gradient similarity between sub-block a and sub-block b at the coding block position $(i,\ j)$ and its calculation formula are shown in Equation (8):

$$GSIM(i,j) = \frac{2\sum_i \sum_j G_a(i,j)G_b(i,j) + C}{\sum_i \sum_j [G_a(i,j)]^2 + \sum_i \sum_j [G_b(i,j)]^2 + C} \tag{8}$$

where $G_a(i,j)$ and $G_b(i,j)$ are the gradient amplitudes of coding blocks a and b at positions $(i,\ j)$, respectively, and C is a constant.

Then, we replace the structure-function $Str(i,j)$ in the structural similarity (*SSIM*) with the gradient similarity $GSIM(i,j)$ to gain the gradient-based structural similarity *GSIM*, as shown in Equation (9). The gradient-based structural similarity is applied to the judgment of CU division, and the average gradient structural similarity (*MGSSIM*) of CU is defined. It is calculated by Equation (10), in which x is the number of similarity values.

$$GSSIM(i,j) = L(i,j)C(i,j)GSIM(i,j) \tag{9}$$

$$MGSSIM = \frac{1}{n}\sum_{x=1}^{n} GSSIM(i,j) \tag{10}$$

Considering the similarity of the 4 sub-CUs (CU_1, CU_2, CU_3, CU_4) after the current coding block is divided by QT (as shown in Figure 6), *GSIM* is calculated in pairs to obtain six similarity values, namely $GSSIM_{12}$, $GSSIM_{13}$, $GSSIM_{14}$, $GSSIM_{23}$, $GSSIM_{24}$, $GSSIM_{34}$, respectively. And the mean value of these six values is utilized to obtain the current average gradient structure similarity of CU *MGSSIM*.

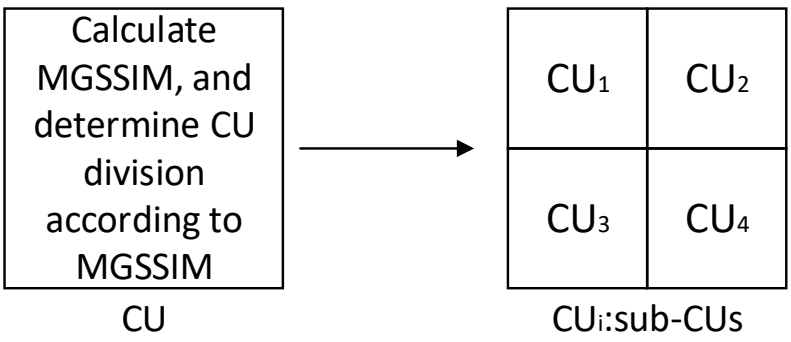

Figure 6. The process of computing MGSSIM.

Since MGSSIM represents the average similarity of the four sub-CUs of the current CU, it can be considered that when the similarity value is less than a certain threshold (set as TH1), that is, the difference between the four sub-CUs is relatively large, it will be more appropriate to divide them into a quadtree. When the similarity value is greater than a certain threshold (set as TH2), the four sub-CUs are extremely similar. If they are coded without partitioning, the residual error and the number of bits used will be smaller and the coding effect will be better.

Threshold Selection

In the proposed CU intra-frame, fast partitioning algorithm based on average gradient structure similarity, we use two thresholds, including TH1 and TH2. Statistical analysis of the characteristics of these two thresholds shows that both TH1 and TH2 are related to the quantization parameter (QP). The threshold Formulas (11) and (12) are defined as follows:

$$\mathrm{TH1} = \alpha \times QP \tag{11}$$

$$\mathrm{TH2} = \beta \times QP \tag{12}$$

where α and β are adjustable parameters.

To verify the feasibility of the algorithm, the division characteristics of video sequences with various resolutions are statistically analyzed. The statistical results are shown in Figure 7. The blue mark is the proportion of quadtree division, the orange mark is the proportion of multi-type tree division, and the gray mark is the proportion of no division. To make the algorithm achieve a good balance between encoding time saving and bit rate loss, we finally set α to 0.005, and β to 0.0084, respectively. This section is not mandatory but can be added to the manuscript if the discussion is unusually long or complex.

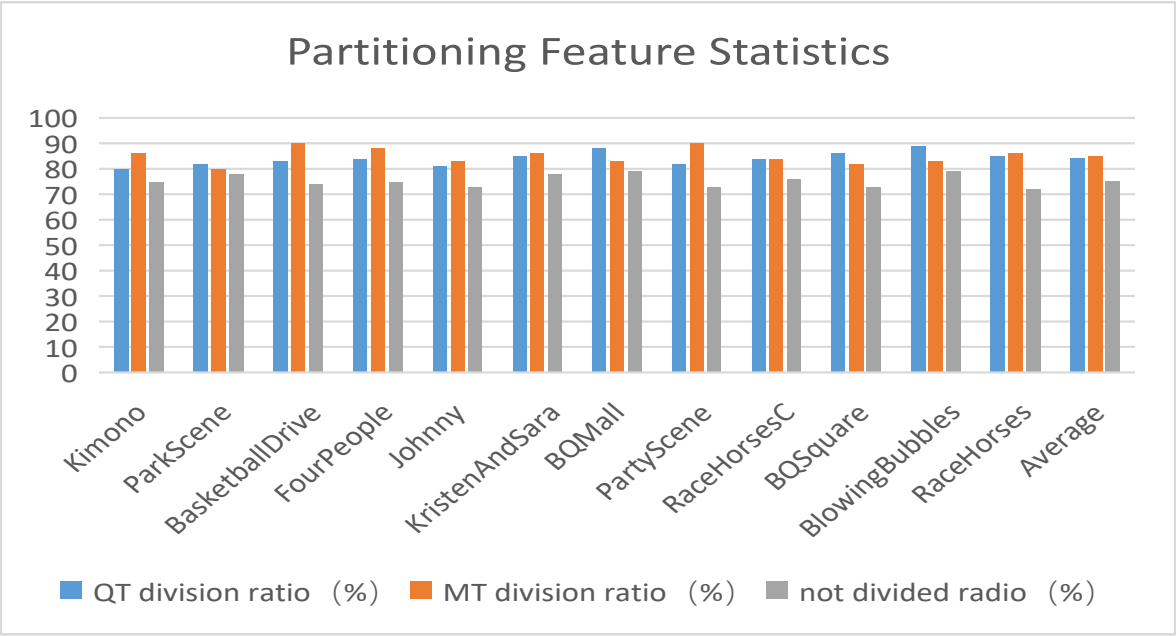

Figure 7. Statistics of the division characteristics of each video sequence.

From the analysis of the table, it can be known that when the *MGSSIM* value of the current CU is greater than TH2, an average of 75.4% of the CUs will terminate the division process early and skip all division methods. When the *MGSSIM* value of the current CU is less than TH1, an average of 83.8% of the CUs are quad-tree divided into smaller CUs. When the *MGSSIM* value of the current CU is between the thresholds TH1 and TH2, an average of 84.5% of the CUs choose to perform binary tree partition or ternary tree partition. From the data in the figure, it can be seen that the gradient

structure similarity has a good performance in the extraction of image details, which also proves the superiority of our designed algorithm.

3.2. Skip Non-Optimal Partition Patterns Based on Standard Deviation

In the above algorithm, we calculated the average gradient structure similarity between CUs, and analyzed their relationship by comparing them with the threshold. To further optimize our algorithm, the standard deviation of the vertical and horizontal directions can be used to further reduce the division mode and achieve the purpose of reducing the encoding time.

When the similarity value is between TH1 and TH2, there are still multiple division candidate modes for the coding block. To make multi-type tree division decisions, we need to judge according to the texture direction of the coding block and skip the impossible division directions. For a vertically textured CU, the luminance difference in the horizontal direction is larger than the luminance difference in the vertical direction. For CUs with horizontal textures, the opposite is true. The standard deviation SD_V and SD_H of its vertical and horizontal features are calculated as:

$$SD_V = \frac{1}{\text{height}} \sum_{i=1}^{\text{height}} \sqrt{\frac{1}{\text{width}} \sum_{j=1}^{\text{width}} pixel(i,j)^2 - \left[\frac{1}{\text{width}} \sum_{j=1}^{\text{width}} pixel(i,j)\right]^2} \tag{13}$$

$$SD_H = \frac{1}{\text{width}} \sum_{j=1}^{W} \sqrt{\frac{1}{\text{height}} \sum_{i=1}^{\text{height}} pixel(i,j)^2 - \left[\frac{1}{\text{height}} \sum_{i=1}^{\text{height}} pixel(i,j)\right]^2} \tag{14}$$

where height and width are the height and width of the CU, respectively, and $pixel(i, j)$ is the pixel value of the current position.

In H.266/VVC, there is a strong correlation between the CU division and the texture information of the image, and the texture direction is related to the change of CU in different directions, so we use the standard deviation of both vertical and horizontal directions compared with the threshold TH3 to determine the texture direction of CU. The calculation expression of $S_{ver/hor}$ is:

$$S_{ver/hor} = \frac{SD_V}{SD_H} \tag{15}$$

The specific expression for texture orientation determination is:

$$S_{ver/hor} \geq \text{TH3} \tag{16}$$

Given that we adopt a fast division of 32 × 32 and 16 × 16 CU, the multi-type tree decision can only be applied to square cells, not non-square cells. If SD_H in a CU is greater than SD_V, then BTH and TTH will be skipped, i.e., for a vertically textured CU, the horizontal partitioning mode will be skipped. If SD_V in a CU is greater than SD_H, then BTV and TTV will be skipped, i.e., for a horizontally textured CU, vertical partitioning mode will be skipped.

Threshold Selection

To distinguish the relationship between the CU partition mode and SD, we selected the VVC standard to specify six video sequences in six types of video test sequences to conduct experiments, including “FoodMarket”, “ParkRunning3”, “Cactus”, “BQMall”, “BasketballPass” and “FourPeople”. The first five frames of each sequence were encoded in all internal (AI) configurations with QP = 27. We collected the statistical results of $S_{ver/hor}$ and the corresponding vertical and horizontal MT partition patterns, overall results as shown in Tables 1 and 2.

Table 1. Statistics of CU partition vertical and horizontal modes when $S_{ver/hor} < 1$.

$S_{ver/hor} < 1$	Vertical Partition Modes (%)	Horizontal Partition Modes (%)
FoodMarket	72	28
ParkRunning3	63	37
Cactus	66	34
BQMall	74	26
BasketballPass	70	30
FourPeople	77	23

Table 2. Statistics of CU partition vertical and horizontal modes when $S_{ver/hor} > 1$.

$S_{ver/hor} > 1$	Vertical Partition Modes (%)	Horizontal Partition Modes (%)
FoodMarket	26	74
ParkRunning3	25	75
Cactus	29	71
BQMall	35	65
BasketballPass	23	77
FourPeople	38	62

The proportions of vertical and horizontal partition patterns in different sequences are shown in the table above. As can be seen from Table 1, when the $S_{ver/hor}$ value is less than 1, the percentage of vertical partitioning mode is 72%, 63%, 66%, 74%, 70%, and 77%, respectively, and the percentage of vertical partitioning mode codes is more than twice that of horizontal partitioning. If the $S_{ver/hor}$ value is greater than the higher threshold TH3, it means that the complexity in the horizontal direction is greater than that in the vertical direction, and the current CU is more likely to have horizontal textures. As can be seen from Table 2, when the $S_{ver/hor}$ value is greater than 1, the average probability of horizontal partitions being encoded is more than twice that of vertical partitions. If the $S_{ver/hor}$ value is less than the lower threshold TH3, it indicates that the current CU is more likely to have vertical textures, so we skip the horizontal partition mode.

The choice of threshold is related to QP. If the TH3 value is too small, the algorithm will get better complexity reduction because more partition patterns are predetermined. If the TH3 value is too large, the encoding performance will be better, but the computational complexity will increase. In the case of QP = 27 and QP = 37, two video sequences "BasketballDrive" and "BQMall" were selected to test the prediction accuracy. We conducted experiments for this purpose, and the relationship between threshold and accuracy is shown in Figure 8. In our method, the overall accuracy of the proposed method is between 80% and 90%. When the QP value is greater than 28, we set the threshold TH3 to 1.7; otherwise, we set the threshold TH3 to 1.2.

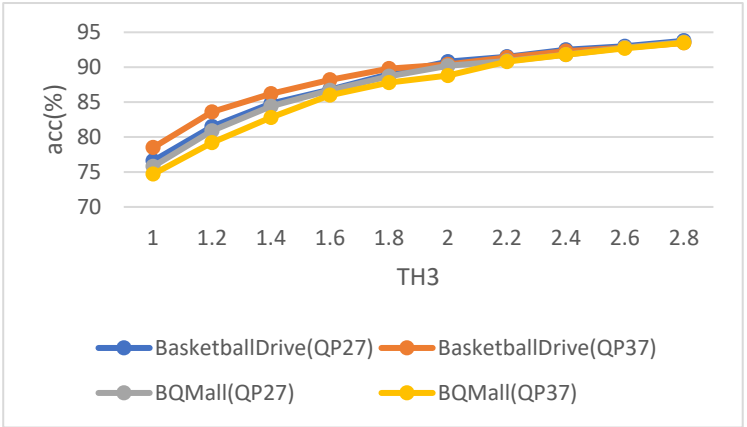

Figure 8. Prediction accuracy at different thresholds.

3.3. The Overall Algorithm

The flowchart is shown in Figure 9, and the specific steps are as follows:

Step 1: Calculate the gradient structure similarity of every 2 CUs in the 4 sub-CUs under the current CU node according to formula (10) to get 6 *GSIM* values, and then take the mean value to obtain the average gradient structure similarity *MGSSIM* of the current CU.

Step 2: When the average gradient magnitude similarity is greater than the threshold TH2, indicating that the four sub-CUs are extremely similar and no tree partition is performed. The intra-prediction modes are directly traversed with the size of the current CU. The same is true for coding, and the recursive traversal of the current CU is ended.

Step 3: If the average gradient structure similarity deviation is less than the threshold TH1, it means that the difference between the four sub-CUs is too large, and the quad-tree division is performed directly to divide the current CU into four smaller CUs, that is, the recursive traversal of the current CU is skipped and the recursive traversal of the four sub-Cu is directly entered.

Step 4: if the average gradient structure similarity is greater than the threshold TH1 and less than the threshold TH2, we make a multi-type tree division decision. The texture direction of the CU is determined according to the standard deviation of the current coding block in different directions. If the standard deviation SD_V in the vertical direction is greater than the standard deviation SD_H in the horizontal direction, which indicates that the texture features of the CU are mainly in the horizontal direction. The current CU can skip subsequent vertical divisions, including vertical binary trees and vertical ternary trees. On the contrary, the current CU can skip the subsequent horizontal binary tree and horizontal ternary tree.

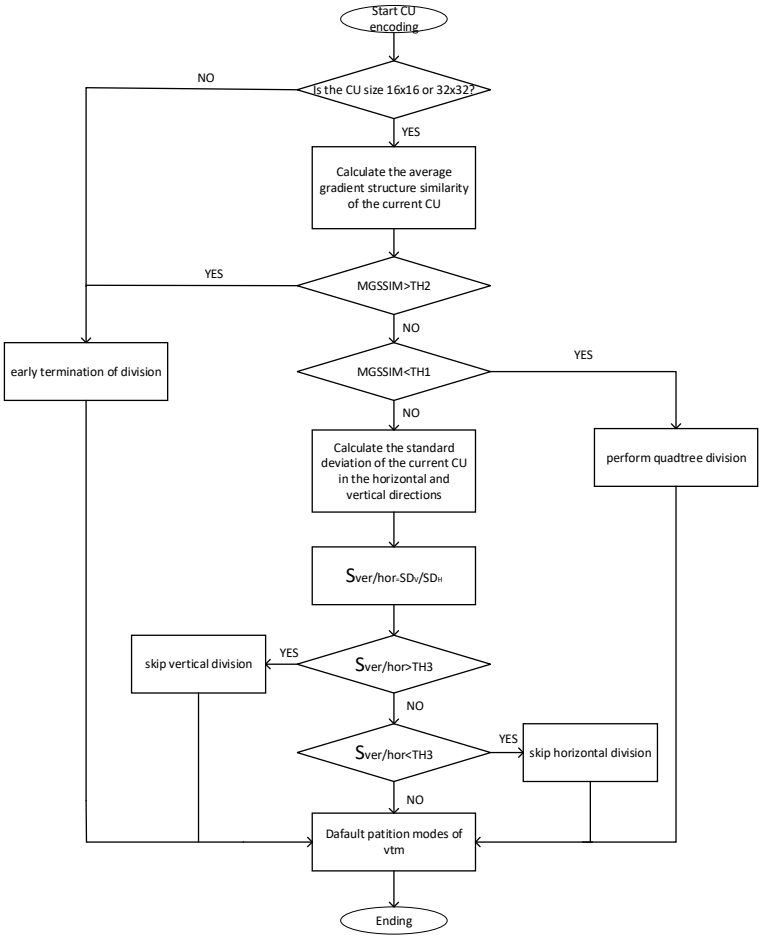

Figure 9. The general framework of the proposed algorithm.

This section is not mandatory but may be added if there are patents resulting from the work reported in this manuscript.

4. Experimental Results

The test conditions and software reference configuration of our proposed algorithm comply with the latest VTM configuration standard. Joint Video Experts Team (JEVT) specifies six types of video test sequences (A1, A2, B, C, D, E) for the VVC standard to test the performance of the algorithm. The video test sequences of various types, different resolutions, and frame rates of VTM standard from 416 × 240 to 3840 × 2160 are selected, and the VTM10.0 model is used for testing. The experimental environment of the test sequence and the configuration parameters of the VTM test model are as follows: processor Intel® Core ™ i5-10500 G CPU, 2.90 GHz main frequency, RAM 8.0 GB, Windows 10 64-bit operating system, and the software development tool is Microsoft Visual Studio 2017. When testing the performance of the algorithm, the standard test video sequences recommended by JVET are encoded in the All Intra (AI) access mode, and the number of encoded frames for each sequence is selected as 100, QP = {22, 27, 32, 37}. The performance of the algorithm is measured by metrics such as the percentage of time saved and the Bjøntegaard incremental bit rate, and the data are averaged under four different QP values. To accurately evaluate the coding performance, this algorithm uses the coding time (ΔT) to evaluate the coding efficiency, and uses the Bjøntegaard incremental bit rate (BDBR) to evaluate the coding quality. The calculation method (17) is:

$$\Delta T = \frac{T_{proposed} - T_{vtm}}{T_{vtm}} \tag{17}$$

Among them, T_{vtm} is the time required for VTM10.0 standard algorithm coding, $T_{proposed}$ is the time required by the improved algorithm, ΔT is the percentage of the difference between the proposed algorithm and the VTM10.0 intra-frame prediction algorithm predicted coding time.

4.1. Comparison with the VTM 10.0 Algorithm

Experiments are carried out according to the above settings, and the comparison results between the algorithm in this paper and the standard algorithm are shown in Table 3. It can be seen that compared with the VTM10.0 standard algorithm, the VVC intra-frame CU partition mode fast decision algorithm proposed in this paper has good coding performance, the average coding time is improved by 49.32%, and the BDBR is only increased by 1.38%. The sequence that saves the most time is BQMall, which saves 56.84% with a 1.46% BDBR increase. The sequence that saves the least time is Kimono, which achieves 38.4% encoding time saving, while only 0.41% BDBR increase. At the same time, the sequence with the largest increase in BDBR is BasketballDrill, and the sequence with the least increase in BDBR is Kimono, which shows that the proposed algorithm is stable and can achieve better complexity reduction on different video sequences.

To further analyze the performance of the proposed algorithm, Figure 10 shows the rate-distortion curves of various sequences using VTM10.0 and the proposed algorithm. The coding curves of the two are almost identical, indicating that the proposed algorithm has almost no decrease in coding performance.

Table 3. Coding performance of the proposed algorithm under VTM10.0.

Sequence Category	Video Sequence	The Proposed Algorithm	
		BDBR (%)	ΔT (%)
A1	Tango2	1.06	45.97
	FoodMarket	1.21	49.62
	Campfire	1.28	51.18
A2	CatRobot1	1.18	46.21
	DaylightRoad2	1.27	49.50
	ParkRunning3	1.32	52.31
B	BasketballDrive	2.28	55.09
	BQTerrace	1.34	52.18
	Cactus	1.29	52.32
	Kimono	0.41	38.4
	ParkScene	0.51	46.31
C	BasketballDrill	2.71	47.36
	BQMall	1.46	56.84
	PartyScene	1.13	48.78
	RaceHorsesC	0.82	48.55
D	BasketballPass	1.42	49.26
	BQSquare	1.54	43.21
	BlowingBubbles	1.37	48.81
	RaceHorses	1.14	44.20
E	FourPeople	1.63	53.36
	Johnny	1.98	51.89
	KristenAndSara	1.93	53.63
	Average	1.38	49.32

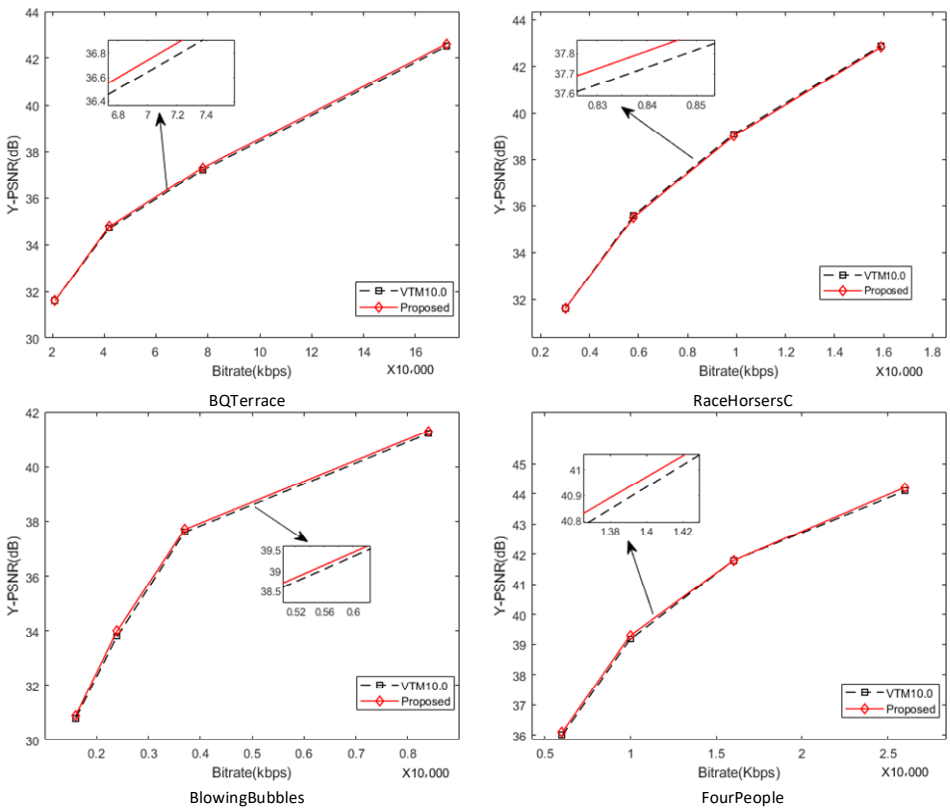

Figure 10. RD curves of VTM10.0 and the proposed algorithm in each sequence.

4.2. Comparison with Other Algorithms

In addition, we also compare the algorithm proposed in this chapter with the existing CU fast partitioning decision algorithm. Considering the different test sequences used by different algorithms, only the same test sequences are selected for comparison to ensure the fairness of the comparison results. In the third algorithm, some experimental results are not given, so they are indicated by "-". Since the human eye is generally not very sensitive to the chromaticity component, the focus here is on the contrast of the luminance component. In order to visually evaluate the performance of different methods, ΔT and BDBR were used as measures in the comparison. The algorithms involved in the comparison include Zhang [17], Zhao [21], and Tang [27], and their results are shown in Table 4. As can be seen from the table, the performance of our algorithm exceeds that of the literature [28] because the time saved by our algorithm far exceeds the performance of this algorithm. The performance of the algorithm in this paper and the algorithm in [17] are not comparable because time and BDBR are mutually constrained. In addition, we compared the proposed algorithm with the existing SVM-based algorithm [21]. With the same test sequence, reference [21] can achieve a time saving of 54.30%, but the BDBR increases by 1.54%. Although a significant saving in coding time can be achieved, the coding performance suffers a significant loss and is not suitable for applications requiring high coding quality.

Table 4. The encoding performance of the proposed algorithm compares with previous works.

Class	Video Sequence	Zhang [17]		Zhao [21]		Tang [27]		Proposed Algorithm	
		BDBR (%)	TS (%)	BDBR (%)	TS (%)	BDBR (%)	TS (%)	BDBR (%)	TS (%)
B	Kimono	1.16	48.76	1.31	54.68	0.87	33.32	0.41	38.40
	ParkScene	1.34	55.93	1.45	48.73	0.83	35.41	0.51	46.31
	BQTerrace	0.88	50.02	1.08	45.30	0.95	34.50	1.34	52.18
C	PartyScene	0.63	49.52	0.87	52.74	0.55	31.10	1.13	48.78
	Race HorsesC	0.71	47.02	1.25	51.07	0.37	23.63	0.82	48.55
	Basket ballDrill	0.96	45.69	1.60	53.92	1.30	33.39	2.71	47.36
D	Blowing Bubbles	0.61	42.07	1.57	51.33	0.95	33.90	1.37	48.81
	Race Horses	0.68	43.28	1.14	55.79	0.71	31.79	0.82	44.20
	BQSquare	0.45	37.14	0.93	54.78	0.68	30.73	1.54	4.21
E	Johnny	-	-	2.37	54.63	-	-	1.98	51.89
	Four People	1.07	54.26	2.19	55.21	1.38	38.01	1.63	53.36
	Kristen AndSara	1.12	52.81	1.88	55.46	1.61	34.84	1.93	53.63
Average		0.87	47.78	1.54	54.30	0.93	32.78	1.35	48.06

In this paper, we study the CU fast partitioning decision algorithm based on three methods and compare the complexity reduction effect with RD performance, and the reduction of coding complexity is the main improvement of our algorithm. As shown in Figure 11. Compared with the literature [17], the algorithm proposed in this paper saves 0.22% in complexity reduction but improves 0.48% in coding loss. Compared with the literature [21], our algorithm is more stable, and although it does not save more coding time, it reduces nearly 0.2% in terms of coding efficiency loss. Compared with the literature [27], the algorithm proposed in this chapter slightly increases the coding efficiency loss but saves more coding time by about 15.28%. Compared with the algorithms proposed in the literature [17,21,27], the coding efficiency is significantly improved, the image quality loss

is smaller, and the overall performance is better. The algorithm in this paper effectively reduces the intra-frame coding time and improves the coding efficiency, while maintaining no significant degradation in code rate and PSNR.

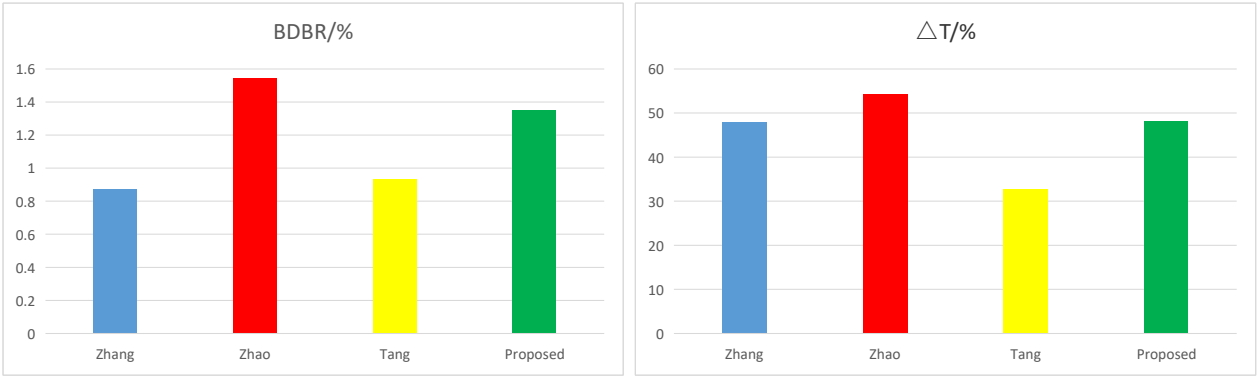

Figure 11. Comparison of experimental results.

5. Summary

This paper proposes a fast coding algorithm for VVC intra-frame based on gradient structure similarity and directional features. First, a new image quality evaluation method—SSIM is introduced in detail, and then the concept of gradient is proposed. The main idea of the algorithm is composed of two main parts. In the first part, given the feature that gradient structure similarity can well extract edge information of coding blocks, the Sobel operator is utilized to extract the gradient features of each sub-CU after the quadtree division, and calculate the average gradient structure similarity between sub-CU. And then the average gradient structure similarity is compared with the threshold value to further judge whether CU terminates partitioning or performs quadtree partitioning. In the second part, considering that there is a strong correlation between the texture direction and CU changes in different directions. The texture direction can be judged by calculating the standard deviation of the current CU vertical direction and horizontal direction, which avoids the unnecessary intra-frame candidate division modes and greatly saves the coding time. The experimental results show that compared with the original platform VTM10.0, the algorithm reduces the encoding time by 49.32% on average while the BDBR only increases by 1.38%, which outperform the state-of-the-art methods.

Author Contributions: Conceptualization, Z.J. and P.L.; methodology, Z.J. and J.Z.; software, P.L.; validation, Z.J., Q.Z. and P.L.; formal analysis, P.L.; investigation, P.L.; resources, Q.Z. and J.Z.; data curation, P.L.; writing—original draft, P.L.; writing—review and editing, Z.J. and J.Z.; visualization, Z.J. and J.Z.; supervision, Q.Z.; project administration, Q.Z.; funding acquisition, Q.Z. All authors have read and agreed to the published version of the manuscript.

Funding: This work was supported in part by the National Natural Science Foundation of China No. 61771432, and 61302118, the Basic Research Projects of Education Department of Henan No. 21zx003, and No. 20A880004, the Key Research and Development Program of Henan No. 222102210156, and the Postgraduate Education Reform and Quality Improvement Project of Henan Province YJS2021KC12 and YJS2022AL034, the key Scientific and Technological Project of Henan Province No. 222102210026, and No. 212102210238, and Henan Key Laboratory of Network Cryptography Technology No. LNCT2021-A15.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Not applicable.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Ye, Y.; He, Y.; Xiu, X. Manipulating ultra-high definition video traffic. *IEEE MultiMedia* **2015**, *22*, 73–81. [CrossRef]
2. Sullivan, G.J.; Ohm, J.; Han, W.; Wiegand, T. Overview of the high efficiency video coding (HEVC) standard. *IEEE Trans. Circuits Syst. Video Technol.* **2012**, *22*, 1649–1668. [CrossRef]
3. Tan, T.K.; Weerakkody, R.; Mrak, M.; Ramzan, N.; Baroncini, V.; Ohm, J.; Sullivan, G.J. Video quality evaluation methodology and verification testing of HEVC compression performance. *IEEE Trans. Circuits Syst. Video Technol.* **2015**, *26*, 76–90. [CrossRef]
4. Filippov, A.; Rufitskiy, V. Recent advances in intra prediction for the emerging h. 266/vvc video coding standard. In Proceedings of the 2019 International Multi-Conference on Engineering, Computer and Information Sciences (SIBIRCON), Novosibirsk, Russia, 21–27 October 2019; pp. 525–530.
5. Chen, J.; Ye, Y.; Kim, S.H. Algorithm description for Versatile Video Coding and Test Model 8 (VTM 8). In *Document JVET-Q1002*; JVET: Brussels, Belgium, 2020.
6. Saldanha, M.; Sanchez, G.; Marcon, C.; Agostini, L. Complexity analysis of VVC intra coding. In Proceedings of the 2020 IEEE International Conference on Image Processing (ICIP), Abu Dhabi, United Arab Emirates, 25–28 October 2020; pp. 3119–3123.
7. Mu, W.; Liang, Y.; Xu, S.; Zhang, W.; Liu, Y. Fast algorithm for HEVC intra-coding implemented by preprocessing. *IET Image Process.* **2019**, *13*, 1578–1586. [CrossRef]
8. Lee, D.; Jeong, J. Fast intra coding unit decision for high efficiency video coding based on statistical information. *Signal Process. Image Commun.* **2017**, *55*, 121–129. [CrossRef]
9. Sun, X.; Chen, X.; Xu, Y.; Wang, Y.; Yu, D. Fast CU partition strategy for HEVC based on Haar wavelet. *IET Image Process.* **2017**, *11*, 717–723. [CrossRef]
10. Du, B.; Siu, W.C.; Yang, X. Fast CU partition strategy for HEVC intra-frame coding using learning approach via random forests. In Proceedings of the 2015 Asia-Pacific Signal and Information Processing Association Annual Summit and Conference (APSIPA), Hong Kong, China, 16–19 December 2015; pp. 1085–1090.
11. Zhang, M.; Lai, D.; Liu, Z.; An, C. A novel adaptive fast partition algorithm based on CU complexity analysis in HEVC. *Multimed. Tools Appl.* **2019**, *78*, 1035–1051. [CrossRef]
12. Zhang, Y.; Li, N.; Kwong, S.; Jiang, G.; Zeng, H. Statistical early termination and early skip models for fast mode decision in HEVC INTRA coding. *ACM Trans. Multimed. Comput. Commun. Appl. (TOMM)* **2019**, *15*, 1–23. [CrossRef]
13. Jiménez-Moreno, A.; Martínez-Enríquez, E.; Díaz-De-María, F. Bayesian adaptive algorithm for fast coding unit decision in the High Efficiency Video Coding (HEVC) standard. *Signal Process. Image Commun.* **2017**, *56*, 1–11. [CrossRef]
14. Sun, X.; Chen, X.; Xu, Y.; Xiao, Y.; Wang, Y.; Yu, D. Fast CU size and prediction mode decision algorithm for HEVC based on direction variance. *J. Real-Time Image Process.* **2019**, *16*, 1731–1744. [CrossRef]
15. Xu, M.; Li, T.; Wang, Z.; Deng, X.; Yang, R.; Guan, Z. Reducing complexity of HEVC: A deep learning approach. *IEEE Trans. Image Process.* **2018**, *27*, 5044–5059. [CrossRef] [PubMed]
16. Chen, K.; Zeng, X.; Fan, Y. CNN oriented fast CU partition decision and PU mode decision for HEVC intra encoding. In Proceedings of the 2018 14th IEEE International Conference on Solid-State and Integrated Circuit Technology (ICSICT), Qingdao, China, 31 October–3 November 2018; pp. 1–3.
17. Zhang, Q.; Zhao, Y.; Jiang, B.; Huang, L.; Wei, T. Fast CU Partition Decision Method Based on Texture Characteristics for H.266/VVC. *IEEE Access* **2020**, *8*, 203516–203524. [CrossRef]
18. Lei, M.; Luo, F.; Zhang, X.; Wang, S.; Ma, S. Look-ahead prediction based coding unit size pruning for VVC intra coding. In Proceedings of the 2019 IEEE International Conference on Image Processing (ICIP), Taipei, Taiwan, 22–25 September 2019; pp. 4120–4124.
19. Saldanha, M.; Sanchez, G.; Marcon, C.; Agostini, L. Fast partitioning decision scheme for versatile video coding intra-frame prediction. In Proceedings of the 2020 IEEE International Symposium on Circuits and Systems (ISCAS), Seville, Spain, 12–14 October 2020; pp. 1–5.
20. Cui, J.; Zhang, T.; Gu, C.; Zhang, X.; Ma, S. Gradient-Based Early Termination of CU Partition in VVC Intra Coding. In Proceedings of the 2020 Data Compression Conference (DCC), Snowbird, UT, USA, 24–27 March 2020; pp. 103–112.
21. Zhao, J.; Wu, A.; Zhang, Q. SVM-Based Fast CU Partition Decision Algorithm for VVC Intra Coding. *Electronics* **2022**, *11*, 2147. [CrossRef]
22. Zhang, Q.; Wang, Y.; Huang, L.; Jiang, B.; Wang, X. Fast CU partition decision for H. 266/VVC based on the improved DAG-SVM classifier model. *Multimed. Syst.* **2021**, *27*, 1–14. [CrossRef]
23. Chen, F.; Ren, Y.; Peng, Z.; Jiang, G.; Cui, X. A fast CU size decision algorithm for VVC intra prediction based on support vector machine. *Multimed. Tools Appl.* **2020**, *79*, 27923–27939. [CrossRef]
24. Huang, Y.H.; Chen, J.J.; Tsai, Y.H. Speed Up H. 266/QTMT Intra-Coding Based on Predictions of ResNet and Random Forest Classifier. In Proceedings of the 2021 IEEE International Conference on Consumer Electronics (ICCE), Las Vegas, NV, USA, 10–12 January 2021; pp. 1–6.
25. Yang, H.; Shen, L.; Dong, X.; Ding, Q.; An, P.; Jiang, G. Low-complexity CTU partition structure decision and fast intra mode decision for versatile video coding. *IEEE Trans. Circuits Syst. Video Technol.* **2019**, *30*, 1668–1682. [CrossRef]
26. Zhang, Q.; Wang, Y.; Huang, L.; Jiang, B. Fast CU Partition and Intra Mode Decision Method for H.266/VVC. *IEEE Access* **2020**, *8*, 117539–117550. [CrossRef]
27. Tang, G.; Jing, M.; Zeng, X.; Fan, Y. Adaptive CU split decision with pooling-variable CNN for VVC intra encoding. In Proceedings of the 2019 IEEE Visual Communications and Image Processing (VCIP), Sydney, NSW, Australia, 1–4 December 2019; pp. 1–4.

28. Tissier, A.; Hamidouche, W.; Vanne, J.; Galpin, F.; Menard, D. CNN oriented complexity reduction of VVC intra encoder. In Proceedings of the 2020 IEEE International Conference on Image Processing (ICIP), Abu Dhabi, United Arab Emirates, 25–28 October 2020; pp. 3139–3143.
29. Li, T.; Xu, M.; Tang, R.; Chen, Y.; Xing, Q. DeepQTMT: A deep learning approach for fast QTMT-based CU partition of intra-mode VVC. *IEEE Trans. Image Process.* **2021**, *30*, 5377–5390. [CrossRef]
30. Park, S.; Kang, J.W. Fast multi-type tree partitioning for versatile video coding using a lightweight neural network. *IEEE Trans. Multimed.* **2020**, *23*, 4388–4399. [CrossRef]
31. Li, Y.; Li, L.; Fang, Y.; Peng, H.; Ling, N. Bagged Tree and ResNet-Based Joint End-to-End Fast CTU Partition Decision Algorithm for Video Intra Coding. *Electronics* **2022**, *11*, 1264. [CrossRef]
32. Wang, Z.; Bovik, A.C. A universal image quality index. *IEEE Signal Process. Lett.* **2002**, *9*, 81–84. [CrossRef]
33. Wang, Z.; Bovik, A.C.; Sheikh, H.R.; Simoncelli, E.P. Image quality assessment: From error visibility to structural similarity. *IEEE Trans. Image Process.* **2004**, *13*, 600–612. [CrossRef] [PubMed]
34. Liu, A.; Lin, W.; Narwaria, M. Image quality assessment based on gradient similarity. *IEEE Trans. Image Process.* **2011**, *21*, 1500–1512. [PubMed]