electronics

MDPI

Article

Efficient CU Decision Algorithm for VVC 3D Video Depth Map Using GLCM and Extra Trees

Fengqin Wang, Zhiying Wang and Qiuwen Zhang *

College of Computer and Communication Engineering, Zhengzhou University of Light Industry, Zhengzhou 450002, China; 2005017@zzuli.edu.cn (F.W.); 332107040640@email.zzuli.edu.cn (Z.W.)
* Correspondence: 2012032@zzuli.edu.cn; Tel.: +86-371-8660-9559

Abstract: The new generation of 3D video is an international frontier research hotspot. However, the large amount of data and high complexity are core problems to be solved urgently in 3D video coding. The latest generation of video coding standard versatile video coding (VVC) adopts the quad-tree with nested multi-type tree (QTMT) partition structure, and the coding efficiency is much higher than other coding standards. However, the current research work undertaken for VVC is less for 3D video. In light of this context, we propose a fast coding unit (CU) decision algorithm based on the gray level co-occurrence matrix (GLCM) and Extra trees for the characteristics of the depth map in 3D video. In the first stage, we introduce an edge detection algorithm using GLCM to classify the CU in the depth map into smooth and complex edge blocks based on the extracted features. Subsequently, the extracted features from the CUs, classified as complex edge blocks in the first stage, are fed into the constructed Extra trees model to make a fast decision on the partition type of that CU and avoid calculating unnecessary rate-distortion cost. Experimental results show that the overall algorithm can effectively reduce the coding time by 36.27–51.98%, while the Bjøntegaard delta bit rate (BDBR) is only increased by 0.24% on average which is negligible, all reflecting the superior performance of our method. Moreover, our algorithm can effectively ensure video quality while saving much encoding time compared with other algorithms.

Keywords: VVC 3D video; depth map; GLCM; Extra trees

Citation: Wang, F.; Wang, Z.; Zhang, Q. Efficient CU Decision Algorithm for VVC 3D Video Depth Map Using GLCM and Extra Trees. *Electronics* **2023**, *12*, 3914. https://doi.org/10.3390/electronics12183914

Academic Editors: Radu Ciprian Bilcu and Ionut Schiopu

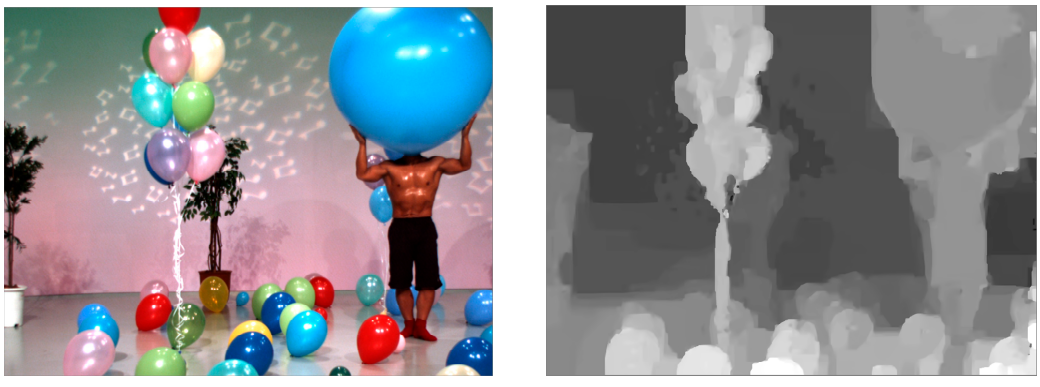
Received: 23 August 2023
Revised: 10 September 2023
Accepted: 13 September 2023
Published: 17 September 2023

1. Introduction

Due to the swift progress in both video coding and multimedia information technology, people's requirements for video quality have gradually increased. The pursuit of temporal and spatial resolution has become increasingly high. Blu-ray, HDR (high dynamic range) and other image quality [1] have been the standard configuration of significant video platforms. On the other hand, people put forward higher requirements for video, such as a more realistic visual experience, as well as a more realistic and natural 3D visual experience. 2D video is brutal in meeting these new needs, so with a large viewing angle, a high quality and picture, surrounded by the sense of the immersive video, came into being. This mainly included stereoscopic video, multi-vision point video, 360° video, VR (virtual reality) video, and AR (augmented reality) video, etc. [2]. The 3D video also shows broad application prospects within medicine, education, and entertainment. The visual experience it brings occupies a place in theaters and home entertainment and has captured the love of many users. At the same time, to meet people's various expectations for video, 3D video needs to use ultra-high-resolution cameras to shoot scenes from multiple viewpoints and encode and transmit the acquired ultra-high-definition video signals [3]. In addition to transmit multiple viewpoint texture videos, an additional depth map needs to be transmitted to generate a virtual new viewpoint, which causes a sharp increase in storage resources and transmission bandwidth. Therefore, how to better save the bit rate of 3D video transmission while ensuring a high-quality visual experience has become a hot topic for domestic and foreign researchers.

Faced with people's new requirements for video quality, the Moving Picture Experts Group (MPEG) and the Video Coding Expert Group (VCEG) officially announced the latest 3D video coding standard, 3D-HEVC (3D-High Efficiency Video Coding), in February 2015, which was a major advancement in the foundation of multi-view video [4,5]. The 3D-HEVC standard utilizes different coding techniques for the main viewpoint and auxiliary viewpoints, while the traditional HEVC coding technique is mainly used for the main viewpoint. In contrast, some new techniques are used for the coding of non-basic viewpoints and depth maps, which are mainly used for the removal of redundant information such as the correlated redundancy between the texture map and the depth map as well as the spatial redundancy inside the depth map [6,7]. This greatly improves the compression and coding efficiency of 3D-HEVC [8]. In terms of traditional video coding standards, the Joint Video Experts Team (JVET) announced the latest video coding standard VVC in July 2020, which has a more efficient coding performance and broad market prospects than the previous generation of video coding standard HEVC with good network adaptability, compression efficiency, etc. [9]. Moreover, VVC adopts a hybrid coding technology framework, and the CU partitioning adopts the QTMT partition structure, which is more diversified and flexible and can be more effectively adapted to high-resolution image coding and decoding processing. In addition, VVC extends the original HEVC encoder on video data by adding new model prediction modes.

Unlike 2D video, 3D video adds multiple viewpoint texture videos and corresponding depth maps. The depth camera acquires the depth map to determine the distance between the object and the camera and map the distance to a grayscale image. Figure 1 shows a frame from the 3D video test sequence "balloons", from which it can be seen that the texture map is more detailed. In contrast, the depth map has more flat areas and sharp edge areas due to the phenomenon of gray level jumps that cause the edges to be unsmooth. The edge areas critically impact the depth map coding quality compared to the large flat areas. The depth map coding technique is implemented to enhance the precision in encoding the edge regions of the depth map in contrast to the extensive flat areas. This plays a pivotal role in determining the quality of that. While VVC 3D video l still uses the QTMT partition structure [10], the partition size and depth of the CU are strongly correlated with the edge features of the depth map [11]. Typically, flat regions are allocated to larger-sized, shallower-depth CU, whereas regions with sharp edges are often divided into smaller-sized, deeper-depth CU. The significant rise in coding complexity and coding time can be attributed predominantly to the flexible QTMT partition structure of VVC 3D video and the incorporation of the depth map coding technique. The current research methods of VVC are primarily used to address the problems associated with 2D video coding, and few are aimed at 3D video coding [12]. Based on this, it becomes imperative to discover a rapid coding algorithm capable of efficiently diminishing the complexity associated with depth map coding of VVC 3D video, all while upholding video quality.



(a) (b)

Figure 1. A frame within the balloons sequence; (**a**) texture map; (**b**) depth map.

To efficiently alleviate the complexity associated with depth map coding, this paper makes the following subsequent contributions on the attributes of the depth map and VVC partitioning. (1) Propose a GLCM-based edge detection algorithm to classify CUs in the depth graph into flat blocks and complex edge blocks, skipping the CU partition of flat blocks to reduce unnecessary rate-distortion optimization (RDO) processes. (2) Propose an Extra trees-based CU partition decision algorithm, which utilizes the Extra trees model to judge the partition mode of complex edge block CUs in advance, avoiding calculating those RD costs with no partition possibilities, thus significantly reducing the computational complexity.

The remainder of this paper is organized as follows. Section 2 furnishes an overview of previous research efforts centered on alleviating the intrinsic complexity associated with 3D video coding. Section 3 presents the proposed fast intra decision algorithm for depth map. Section 4 showcases the empirical findings and analysis of the comprehensive algorithm put forth in this paper. Finally, the paper concludes with a summary presented in Section 5.

2. Related Works

In recent years, video coding technology has developed rapidly, and many researchers have proposed several fast algorithms for two-dimensional video coding. For example, ref. [13] introduced four algorithm adjustments and a fully parallel hardware architecture for the H.265/HEVC intra encoder. Ref. [14] proposed a new end-to-end fast algorithm to help the coding tree unit partition structure decision in intra coding to reduce the complexity in VVC coding. Ref. [15] used a fast decision algorithm based on the DenseNet network and decision tree classifier to reduce VVC encoding time.

Due to the differences between 2D video and 3D video, the fast coding algorithms applicable to 2D video are unsuitable for use in VVC 3D video. In contrast, the current fast algorithms for reducing the coding complexity for 3D video can be categorized into three types, described next.

2.1. Fast Algorithm Based on Heuristic

Most of the heuristic-based fast algorithms are implemented by using thresholds, RD costs, or correlations between time/space/viewpoints and the characteristics of the video itself. Li et al. [16] proposed a fast algorithm utilizing spatial correlation and RD cost to diminish the complexity of intra coding for the depth map and harnessed the frequency distribution attributes of RD cost to accurately predict the maximal depth layer of CTUs. Moreover, to diminish the complexity of depth modeling mode 1 (DMM1), a fast decision method for wedge patterns based on K-means was proposed. Ref. [17] proposed a two-layer texture discriminative fast depth coding algorithm, which uses the summation of gradient matrices to compute the texture intricacy of CU and their sub-blocks within the present depth map to determine further the depth of CU partition as well as to skip unnecessary DMM discriminations. Fu et al. in [18] proposed a two-step adaptive corner selection technique using the feature of corners in computer vision, analyzed and investigated the correlation between corners and coding modes, and proposed a fast intra mode decision for non-corner PU that can skip the DMM decision and feel the segmented depth coding in advance. Ref. [19] proposed a fast algorithm for diminishing complexity based on texture features and spatiotemporal correlation, using a combination of pixel-based statistical methods and edge detection to establish a new texture complexity model, based on which CUs are categorized into smooth, texture, and edges. After that, an early termination of the CU partitioning algorithm for smoothing blocks was proposed aiming to eliminate superfluous CUs. Hamout et al. [20] proposed a method to extract the edge information of each PU pixel by performing texture and depth maps using a local structure tensor and then building a histogram of the edge directions and opting for the orientation characterized by the highest histogram value as the optimal intra prediction pattern.

2.2. Fast Algorithm Based on Machine Learning

Integrating machine learning technology has substantially improved the efficiency of 3D video coding and effectively reduced the intricacy. Li et al. [21] proposed an early intra adaptive CU decision algorithm for unsupervised learning-based intra depth map coding, which proposed three clustering models for different sizes of CUs to determine in advance whether the CUs need to be further classified. Ref. [22] proposed a fast XGBoost-based algorithm to select a large number of features for model training by using the correlation between spatiotemporal, inter-view, etc., and a comprehensive set of 14 XGBoost models was constructed to address various CU sizes and viewpoint types, aiming to achieve the determination of the early CU partition and the prediction of PU. Saldanha et al. [23] proposed a fast algorithm for depth map coding based on static decision trees, which constructs different decision trees for three sizes of CU, utilizing data mining and machine learning. It extracts context attributes to determine whether the CUs need to be segmented, which effectively avoids the complex RDO process. To more effectively address the challenge posed by the elevated complexity of the depth map, ref. [24] proposed a fast CU decision making for the depth map based on the XGBoost model; first of all, the decision model is constructed to use the texture information of the depth map as a feature attribute vector, simultaneously, the determination of whether the current CU should undergo further partitioning is designated as the label. In addition, the feature attributes obtained from the coding process are utilized to train the decision model and to determine whether the CU continues to be splited or not. An early determination algorithm for deep intra coding was proposed by FU et al. in their work cited in [25], where the intra 2N $\times$ 2N and N $\times$ N and CU partitions, of whether or not the current CU needs to be skipped or examined, are regarded as binary decisions; following the execution of the decision tree based learning algorithm, the Gini values are obtained from the leaf nodes for each of the partitions proposed for the current CU, while the results are restricted by different Gini values.

2.3. Fast Algorithm Based on Deep Learning

Over recent years, deep learning has gained extensive adoption across numerous domains, including its application within video coding. Some practical, fast algorithms are based on deep learning in 3D video coding. Peng et al. in [26] proposed a deep loop filtering method based on multidomain correlation learning and partition constraint networks to improve the performance of multi-view video coding by exploring the multidomain correlation to recover the high-frequency details of the distorted frames as well as designing the partition loss and thus the compression artefacts for better attenuation. To alleviate the complexity associated with depth map coding, a fast depth map intra coding algorithm employing CNN and layer classification is proposed in [26], which consists of a layer classification model for texture smoothing to ascertain the most smoothed depth map and a CNN network that contains the SENet structure that combines these two models in order to predict the delineation of all the CU under a particular view. To enhance the efficacy of intra prediction in depth map coding, Zhang et al. proposed an intra prediction mode based on depth region partition [27], introducing a depth region partition network and applying it to texture frames to directly predict its division results. Furthermore, a frame-level training strategy is devised for the information edge representation. Xie et al. [28] proposed a CNN-based edge detection system based on the idea of FNN and deep supervised networks, which showed promising results in edge detection by initializing the network structure and parameters using a pre-trained constructed VGGNet and by amalgamating visual responses across multiple scales and levels.

3. Proposed Algorithm

3D video is encoded with inputs of multi-viewpoint videos, which are simultaneously captured by multiple cameras in the same scene and from different angles. Hence, each viewpoint contains a texture video and a depth video. Therefore, for 3D video, not only the texture video needs to be encoded, but also the corresponding depth map, which

significantly contributes to the elevated complexity observed in 3D video coding. In contrast, VVC 3D video uses the VVC video coding technique, calculated by recursively partitioning and traversing the CU blocks. Therefore, if the depth map is coded directly using the VVC video coding technique, each CU has six possible partition modes, which requires a complete RDO search, traversing and calculating the RD cost of partition modes for all depths, and finally selecting the partition mode with the minimum RD cost for the current CU. The whole process is very cumbersome, which is an essential reason for the high coding complexity.

3.1. Edge Complexity Detection Algorithm Based on GLCM

Predominantly, the depth map comprises sizable flat regions interspersed with smaller segments exhibiting intricate edge patterns [29]. As depicted in Figure 2, the pixel values in the flat areas have characteristics of regional consistency. Within a specific range, the pixel values change relatively slowly and tend to be the interior or background part of the object, so they are split mainly by large-size CU. In contrast, the sharp and complex edge regions tend to be the object's outline, the pixel values change significantly, and the regions are presented with sharp edges. They are primarily divided into small-size CU. Chen et al. [30] proposed a fast GLCM-based algorithm in order to effectively reduce the depth map coding complexity in 3D-HEVC, which can effectively describe the texture complexity of the depth map in 3D-HEVC and thus pre-determine the CU partition depth and candidate intra prediction modes. To efficiently mitigate the coding complexity associated with the depth map of VVC 3D video, the GLCM and Sobel operator extract texture features and edge information from the CU within the depth map and classify the CU blocks into flat blocks, complex blocks, and edge blocks. For a CU classified as a flat block, its subsequent partition will be skipped; for a CU classified as a complex block or an edge block, its division method will be further judged.

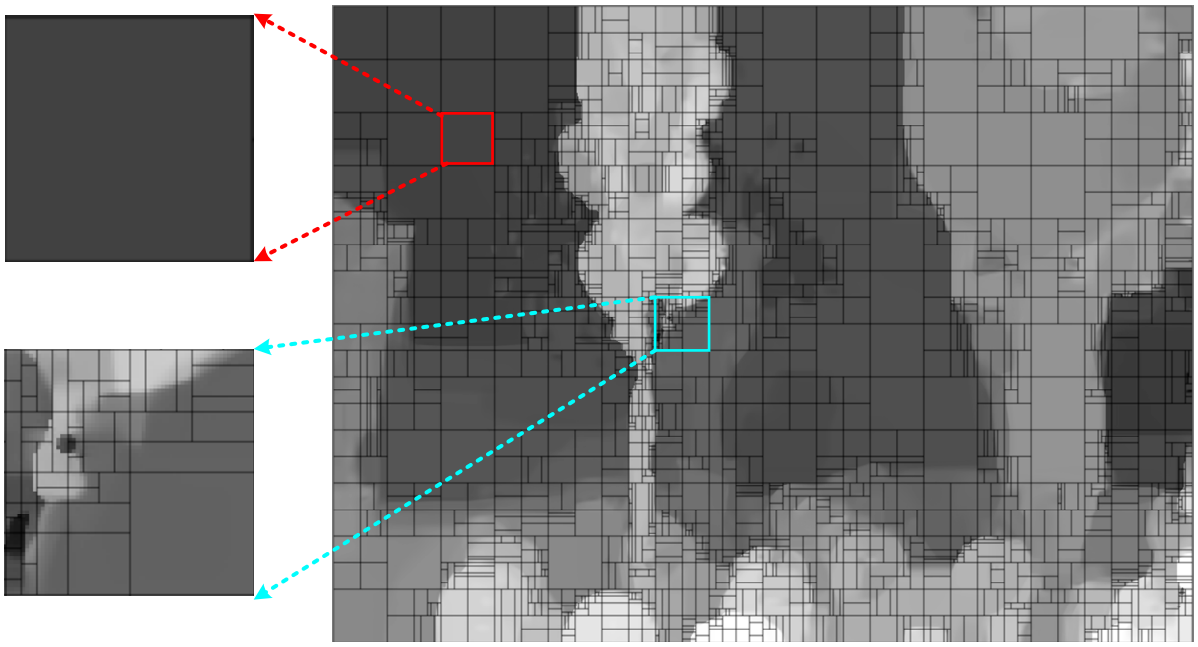


Figure 2. Partition of balloons sequence in depth map in VTM 10.0.

For an image with gray level n, its gray co-production matrix is an $n \times n$ two-dimensional matrix, each element in the matrix represents a second-order joint probability, and the element $p(i, j, d, \theta)$ located at matrix (i, j) represents the probability of the occurrence of a set of pixel pairs (i, j), spaced at a distance of d pixels along the direction θ in the image. Figure 3 shows the generation process of the GLCM of an image block with a size of 4×4 and gray level $n = 4$; (a) represents the pixels in the image block, and (b) represents

the corresponding generated GLCM, where $d = 1$, $\theta = 0^{\circ}$, that is, considering horizontally adjacent pixel pairs. In the figure, the horizontally adjacent pixel pair (2,3) occurs twice, so the element at the (2,3) position in the obtained GLCM is 2, while the pixel pair (1,1) occurs only once, so the element at (1,1) in the obtained GLCM is 1. According to the representation method of the GLCM in [31], given an image I with M × N resolution units whose gray scale ranges from 0 to $L-1$, the image I can be represented as $\mathrm{I} = [f(i,j)]MN$, where $f(i,j)$ is the gray scale value at point (i,j). The GLCM can be represented as follows:

$$\begin{matrix} P(i,j,d,\theta)=\#\{(k,l),(m,n)\in M\times N,\ f(k,l)=i,f(m,n)=j,dis((k,l),(m,n))=d\} \\ 0\le i,j\le L-1, \\ \theta\in\{0^{\circ},45^{\circ},90^{\circ},135^{\circ}\} \end{matrix}, \tag{1}$$

where $dis()$ is the distance function to denote the distance d between (k,l) and (m,n), $f(k,l) = i$ and $f(m,n) = j$ represent the gray values of points (k,l) and (m,n), respectively, # represents the number of elements in the set. The normalized GLCM is obtained by dividing each entry by the count of adjacent resolution pixel pairs R:

$$p(i,j,d,\theta) = \frac{P(i,j,d,\theta)}{R}, \tag{2}$$

0	1	2	2	
2	3	1	1	
1	2	3	1	
2	4	0	2	

(a)

	1	2	3	4
1	1	2	0	0
2	0	1	2	1
3	2	0	0	0
4	0	0	0	0

(b)

Figure 3. GLCM generation diagram; (**a**) pixel matrix, (**b**) GLCM matrix.

Haralick defines a set of 14 texture feature measures based on the GLCM [32], and here we choose three of them, which are angle-second matrix (ASM), contrast (*CON*), and correlation (*COR*).

The ASM represents the summation of element squares within the matrix. It is commonly utilized to gauge the homogeneity of grayscale values along a specific texture direction in an image, as well as to quantify the consistency of texture grayscale variations. The expression for *ASM* is provided below:

$$ASM = \sum_{i=0}^{m-1}\sum_{j=0}^{n-1} p(i,j,d,\theta)^2, \tag{3}$$

The *CON* metric quantifies the intensity variation between a pixel and its neighboring pixels across the entire image. It is defined as follows:

$$CON = \sum_{i=0}^{m-1}\sum_{j=0}^{n-1} (i-j)^2 p(i,j,d,\theta), \tag{4}$$

Correlation (*COR*) is used to measure the correlation dependence of gray scale image values between rows or columns of pixels, each CU of the depth map generates a gray scale covariance matrix for four directions $\theta \in \{0^{\circ}, 45^{\circ}, 90^{\circ}, 135^{\circ}\}$. *COR* can be expressed as follows:

$$COR = \sum_{i=0}^{m-1}\sum_{j=0}^{n-1} \frac{(i-\mu_{\mathrm{h}})(j-\mu_v)p(i,j,d,\theta)}{\sigma_h\sigma_v}, \tag{5}$$

where i and j denote the positions of the matrix elements, μ_v and σ_v denote the mean and standard deviation of matrix elements in the vertical direction, and μ_{h} and σ_h represent the mean and standard deviation of matrix elements in the horizontal direction, respectively:

$$\mu_{\mathrm{h}} = \sum_{i=0}^{m-1} \sum_{j=0}^{n-1} ip(i,j,d,\theta), \tag{6}$$

$$\mu_v = \sum_{i=0}^{m-1} \sum_{j=0}^{n-1} jp(i,j,d,\theta), \tag{7}$$

$$\sigma_h = \sqrt{\sum_{i=0}^{m-1} \sum_{j=0}^{n-1} (i-\mu_v)^2 p(i,j,d,\theta)}, \tag{8}$$

$$\sigma_v = \sqrt{\sum_{i=0}^{m-1} \sum_{j=0}^{n-1} (j-\mu_h)^2 p(i,j,d,\theta)}, \tag{9}$$

It is well known that most images have gray levels between 0 and 255, which makes the computation of the GLCM very complicated, and therefore the pixels are properly controlled to quantize the distance function $dis()$ during the computation so that $dis() \in [1, M-1]$. At this point it will make the weaker edges produce a small fraction of distortion, so we use the Sobel operator to solve this problem according to the method in [33]. In the Sobel operator, G_h and G_v represent the gradient of each pixel in the horizontal and vertical directions, and the final gradient of each pixel is as below:

$$|G| = |G_h| + |G_v|, \tag{10}$$

When the magnitude of $|G|$ exceeds the threshold, the pixel is categorized as an edge point. As shown in Figure 4 for the CU of flat region, the pixels typically exhibit similar values in the four directions $(0°, 45°, 90°, 135°)$, $GFV_\theta = (0, 1, 0)$, and define the GLCM feature vector of CU as follows:

$$GFV_\theta = (ASM, CON, COR), \tag{11}$$

$$\overline{GFV} = \frac{GFV_{0°} + GFV_{45°} + GFV_{90°} + GFV_{135°}}{4}, \tag{12}$$

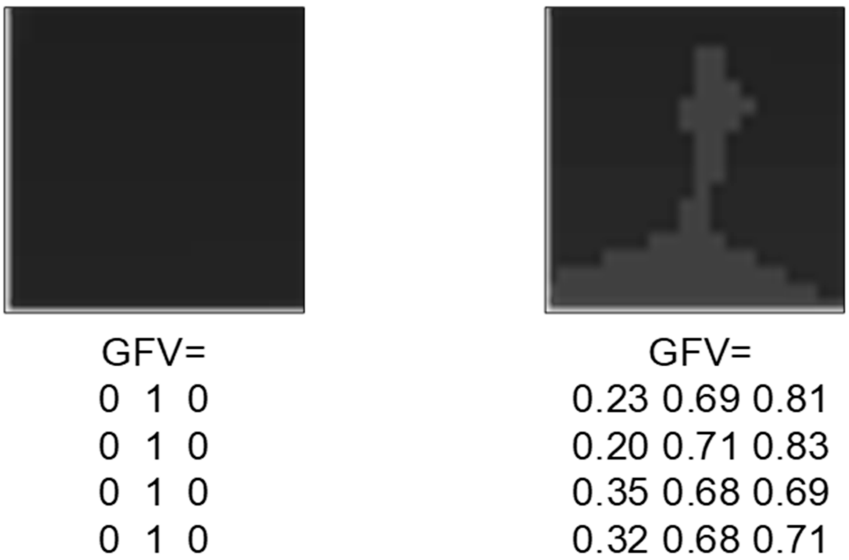


Figure 4. GFV of flat area and complex edge area.

After that, we judge the CU texture classification according to Equation (13), and if the current CU meets the determination condition of the flat texture region, the CU partition will be terminated early; if the present CU is determined as an edge region or a complex region, the CU will be encoded subsequently:

$$\text{Texture}_{CU} = \begin{cases} smooth, \overline{GFV} = (0,1,0) \; and \; |G| < Th \\ edge, |G| > Th \\ complex, \overline{GFV} \neq (0,1,0) \; || \; |G| > Th \end{cases}, \tag{13}$$

3.2. CU Fast Decision Algorithm Based on Extra Trees

Recently, machine learning has found application across diverse domains to improve efficiency [34], among which ensemble methods such as bagging and boosting have received much attention from researchers because they can be used to design efficient classifiers. Among these methods, ensembles of decision trees, such as random forests, which are a series of classifier ensemble methods that use randomization to produce different individual classifiers, have been shown to outperform methods such as SVM and the boosting on a variety of classification tasks. Extra trees is an algorithm proposed by PierreGeurts et al. in 2006 [35], which is very similar to the principle of random forest, and also uses random sampling and random feature selection to construct multiple decision trees. Nevertheless, the difference is that Extra trees will randomly choose the division point of features when splitting nodes, instead of using the optimal division point like decision trees and random forests; that is to say, Extra trees is more random. Moreover, each decision tree of Extra trees is constructed from the original training samples, which makes it better to reduce the model variance and inhibit overfitting, strengthen the robustness of the model, and minimize the bias generated. Since the node division process of Extra trees is simple and there is no need to search for local optimal division points, the training speed is faster, and the computational complexity is consequently diminished.

VVC adopts the QTMT partition structure, where the MTT (multi-type tree) structure is a newly added partition method, which makes the VVC partition more flexible, including horizontal binary tree (BTH) partition, vertical binary tree (BTV) partition, horizontal trinomial tree (TTH) partition, and horizontal trinomial tree (TTV) partition. If the traditional RDO search process is used, the RD cost of all possible CU will be checked sequentially from the top to the bottom layer. Combining the minimum RD cost will be chosen as the best partition result. A CTU coding process needs to calculate the RD cost of 5781 CU, which substantially increases the VVC coding complexity. The coding process of VVC 3D video, the QTMT partition structure, is employed. According to the partition direction, there are six possible partition modes for CUs, which can be roughly categorized into three kinds of partition modes, namely, QT, MTH, and MTV. Therefore, a fast decision algorithm for CU using the Extra trees mode is proposed to make advance decisions for QT, MTH, and MTV to reduce the unnecessary RDO process.

Given that the largest size of MTT in VVC 3D video is 32 × 32, the algorithm applies to 32 × 32, 16 × 16, 8 × 8, and 4 × 4 CUs in order to make the CU partition prediction more accurate, address the computational complexity of features, and mitigate unnecessary computational overheads. Therefore, all things considered here, block shape ratio (BSR), variance (Var), texture trend (T) in the direction of partition, the difference between the predicted partition depth and the actual depth of the current CU (ΔD) and QP, are used as features for Extra trees model training.

(1) Block Shape Ratio (BSR): The aspect ratio of CUs affects the tendency of partition in different directions, which can represent the measurement of the shape of the CU, and the BSR can be expressed according to the method in [36] as follows:

$$BSR = \begin{cases} \frac{h}{w+h} & MTH \\ 0.5 & QT \\ \frac{w}{w+h} & MTV \end{cases}, \tag{14}$$

where w denotes the width of the CU, and h represents the height of the CU.

(2) Variance (Var): It is the variance of total pixels within the given CU, providing a clear reflection of the image's contrast information. Texture information has a certain relationship with the partition type of CU Var is expressed as below:

$$Var = \frac{1}{w \times h} \sum_{i=0}^{w-1} \sum_{j=0}^{h-1} (P(i,j) - meanP)^2, \tag{15}$$

$$meanP = \frac{1}{w \times h} \sum_{i=0}^{w-1} \sum_{j=0}^{h-1} P(i,j), \tag{16}$$

where $P(i,j)$ represents the original pixel located at coordinates (i,j), and $meanP$ signifies the average value computed across all pixels within the given CU.

(3) Texture tendency of partition direction (T): Both the texture direction and gradient of the CU are intricately linked with the partition method. If the texture of the CU is vertical or the vertical gradient is large, then the partition in the vertical direction will be more inclined; on the contrary, if the texture of the CU is horizontal or the horizontal gradient is large, then the partition in the horizontal direction will be more inclined. Therefore, the texture trend T of the division direction here will be calculated by the gradient, and the gradients in different directions will be obtained by the Scharr operator calculation method in [37]. The Scharr operator can be said to be an improvement of the Sobel operator, but the calculation accuracy is higher, and the effect is better. The Scharr operator of each pixel in different directions is shown in Figure 5. Then the gradient is calculated as follows:

$$grad_x = \sum_{i=1}^{w-2} \sum_{j=1}^{h-2} |A \cdot scharr_x| + \varepsilon, \tag{17}$$

where A is the pixel matrix, x denotes the direction of CU ($x = hor, ver, 45^\circ, 135^\circ$), $scharr_x$ is the matrix of the Scharr operator for the corresponding direction, and ε is valued at 1 here (to prevent smoothing with a denominator of 0 in the calculation of T). Then the formula for dividing the texture trend T in the direction is as follows:

$$\begin{pmatrix} T_{\text{hor}} = min\left(\binom{grad_{hor}}{grad_{ver}}, \sqrt{2}\binom{grad_{hor}}{grad_{45^\circ}}, \sqrt{2}\binom{grad_{hor}}{grad_{135^\circ}}\right) \\ T_{\text{ver}} = min\left(\binom{grad_{ver}}{grad_{hor}}, \sqrt{2}\binom{grad_{ver}}{grad_{45^\circ}}, \sqrt{2}\binom{grad_{ver}}{grad_{135^\circ}}\right) \end{pmatrix}, \tag{18}$$

where $grad_{hor}$, $grad_{ver}$, $grad_{45^\circ}$, and $grad_{135^\circ}$ signify the gradient along the horizontal, vertical, 45°, and 135° directions, correspondingly.

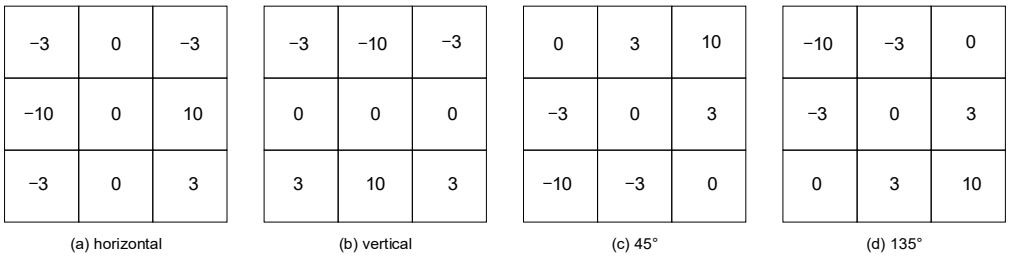


Figure 5. Scharr operators in different directions.

(4) QP: The quantization parameter of the current CU. QP can affect the spatial details of CU partitioning and partitioning results. The larger the QP, the larger the CU partition size; the smaller the QP, the smaller is the CU partition size.

(5) The difference between the predicted partition depth and the real depth of the current CU (ΔD): There exists a relationship between the information of neighboring CU and the current CU partition, and the depth of the current CU partition can be predicted based on the depth of neighboring CU, then ΔD can be expressed as below:

$$\Delta D = D_{pre} - D_{cur}, \tag{19}$$

where D_{pre} denotes the predicted depth and D_{cur} the true depth of the current CU. If only the left (upper) CU exists for the current CU, then the value of D_{pre} is equal to the depth of the left (upper) CU. If both the left and upper CU of the current CU exist, then the value of D_{pre} can be expressed as follows:

$$D_{pre} = \begin{cases} \max\left(D_{upper}, D_{left}\right), D_{l-u} = \min\left(D_{upper}, D_{left}, D_{l-u}\right) \\ \min\left(D_{upper}, D_{left}\right), D_{l-u} = \max\left(D_{upper}, D_{left}, D_{l-u}\right), \\ D_{upper} + D_{left} + D_{l-u}, otherwise \end{cases} \tag{20}$$

where D_{upper}, D_{left}, and D_{l-a} represent the depth of the upper CU, left CU, and upper left CU of the current CU, respectively.

We set the labels of the model to three modes: QT, MTH, and MTV, and set the threshold to 0.2. Table 1 presents the depth map standard test sequence chosen for dataset creation, displaying details such as resolution, frame count, frame rate, and video sequence viewpoint. The test set consists of partial sequences extracted from the JVT-3V standard set, including Kendo, Undo_Dancer, and Poznan_Street. Each sequence was encoded in VTM10.0 with "All intra" configuration. In addition, in order to control the number of Extra trees, the parameter settings for Extra trees are shown in Table 2. In the experiment, the Extra trees model is deployed to VTM10.0 in the "All intra" configuration, and the full sequence test is performed on these two types of 3D video sequences.

Table 1. 3D Video Test Sequence.

Video Sequences	Resolution	3-Views Input	Frame Rate	Frames to Be Encoded
Undo_Dancer		1-5-9	25	250
Poznan_Hall2		7-6-5	25	200
Poznan_Street	1920 × 1088	5-4-3	25	250
Shark		1-5-9	30	300
GT-Fly		9-5-1	25	250
Kendo		1-3-5	30	300
Balloons	1024 × 768	1-3-5	30	300
Newspaper		2-4-6	30	300

Table 2. Parameter settings of the Extra trees model.

Parameter Name	Model Setting
Number of features	5
Min_samples_of leaf	10
Max depth	10
Number of labels	3

3.3. Framework of the Overall Algorithm

Building upon the aforementioned groundwork, our proposed algorithm encompasses two distinct stages of decision processes, including the employment of the edge complexity detection algorithm rooted in the GLCM and the CU fast decision algorithm that utilizes Extra trees, with the aim of diminishing the computational complexity involved in coding of the depth map within VVC 3D video. First, we utilize the GLCM to detect the depth map, classify the CUs in different regions, and skip the CUs in flat blocks, which significantly reduces the RD cost computation and avoids the unnecessary CU delineation process in the intra prediction part. After that, the features are extracted for the CUs classified as complex blocks or edge blocks in the previous stage, and then the Extra trees model is employed for ascertaining the current partition type of the CU and whether the QT partition MTV partition or MTH partition needs to be skipped or not. This can further reduce the RD cost

computation in the CU partition process. The flowchart of the proposed overall algorithm is shown in Figure 6. The pseudocode of the proposed algorithm is shown in Algorithm 1.

Algorithm 1 The proposed fast decision algorithm for VVC 3D video CU split.

Require:
Validity of neighboring frames; the size of the CU input into the Extra trees model is 32 × 32 or 16 × 16 or 8 × 8 or 4 × 4
Ensure:
CU is classified into smooth blocks and complex edge blocks; CU skips unnecessary partitioning types

1: Input: current coding unit
2: Calculate the gradient of the current CU $|G| = |G_h| + |G_v|$ by Equation (10);
3: Calculate the feature vector $\overline{GFV} = \frac{GFV_{0^\circ} + GFV_{45^\circ} + GFV_{90^\circ} + GFV_{135^\circ}}{4}$ by Equation (12).
4: **if** $(\overline{GFV})$ = (0,1,0) **and** $|G|$ < Th **then**
CU is classified as a smooth block and terminate the CU partition;
else if $(\overline{GFV}) \neq$ (0,1,0) or $|G|$ > Th **then**
CU is classified as a complex or edge block;
5: **if** CU == 32 × 32 or 16 × 16 or 8 × 8 or 4 × 4 **then**
6: Compute the block shape ratio, variance, texture trend in the partition direction, the difference between the predicted partition depth and the true depth of the current CU, and QP;
7: Obtain the probabilities of QT, MTH, and MTV through the Extra trees model.
8: **if** probabilities of QT ≤ th **then**
skip QT;
if probabilities of MTH ≤ th **then**
skip MTH;
if probabilities of MTV ≤ th **then**
skip MTV;
9: End.

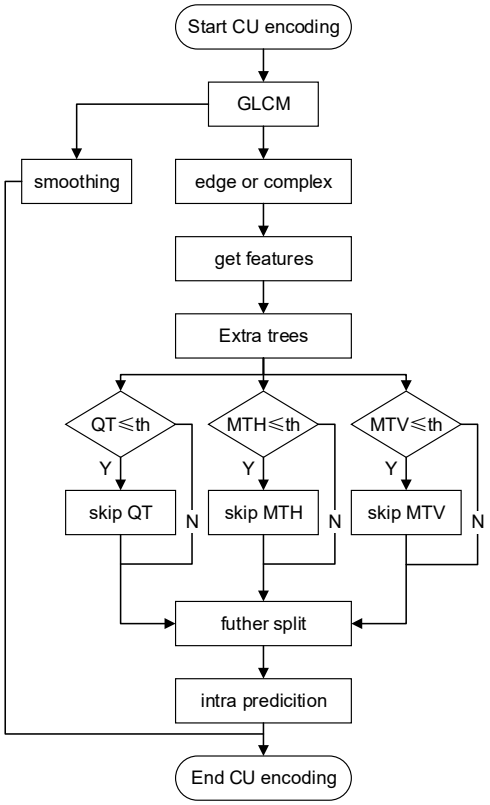


Figure 6. Overall block diagram of the proposed algorithm.

4. Experimental Results

To assess the efficacy of the proposed algorithm in mitigating the computational complexity of VVC 3D video, the proposed scheme was implemented within the reference test software VTM10.0 with ALL-Intra configuration and QP (depth) settings of {34, 39, 42, 45}, executed on an Intel(R) Core (TM) i7-11800H and 16GBRAM platform. Furthermore, it was evaluated on eight video sequences with two types of resolution ($1024 \times 768/1920 \times 1088$) from the 3D standard video test sequences of the JVT-3V, as shown in Table 1. The algorithm proposed in this paper only improves the depth map coding of VVC 3D video and uses BDBR [38] and ΔT as the criteria for evaluating the performance of the proposed algorithm. Among them, BDBR is the coding efficiency saving standard of different methods at the same desired quality level (BDBR indicates the percentage of code rate reduction achievable through an improved coding approach while maintaining an equivalent objective quality. A lower BDBR value signifies enhanced compression performance of the current encoder), and ΔT represents the percentage reduction in coding time when compared to the reference algorithm in VTM10.0, which is defined as below:

$$\Delta T = \frac{T_{PRO} - T_{VTM10.0}}{T_{VTM10.0}} \times 100\%, \tag{21}$$

4.1. Performance Analysis of Individual Algorithm

The proposed overall algorithm consists of the GLCM based edge complexity detection algorithm and Extra trees-based CU fast decision algorithm, where the GLCM-based edge complexity detection algorithm classifies the CUs in the images according to their complexity into flat block CU, edge block, or complex block CU. For the CU that are classified as flat block, their subsequent classification will be skipped; for CU classified as complex blocks and edge blocks, the partition method will be further judged, which can effectively avoid the RDO process for flat areas in the depth map. The Extra trees-based CU fast decision algorithm extracts the relevant features. It utilizes the Extra trees model to assess the partition type of the CUs of the complex and edge blocks that need to be split, avoiding unnecessary RD cost calculation of the partition type. The two sub-algorithms enable the overall algorithm to be applied more effectively to diminish the computational burden of depth map coding of the VVC 3D video.

In this section, we conduct ablation experiments to independently evaluate the effectiveness of the two sub-algorithms. First, the GLCM-based edge complexity detection algorithm is used to classify CUs, and then the VTM10.0 anchoring algorithm is used to split the CU, which needs to go through a tedious RDO process. Then, we remove the first sub-algorithm and directly adopt the Extra trees model to judge the partition type of all CUs. Finally, the BDBR and ΔT of the three combinations are analyzed separately, and Table 3 presents the corresponding experimental results.

Table 3. Comparison of sub-algorithm and overall algorithm encoding performance.

Sequence	GLCM		Extra Trees		Overall	
	BDBR (%)	ΔT (%)	BDBR (%)	ΔT (%)	BDBR (%)	ΔT (%)
Balloons	0.25	23.09	0.24	33.21	0.18	39.45
Kendo	0.12	31.25	0.21	31.04	0.16	36.27
Newspaper	0.24	26.41	0.35	34.72	0.39	43.09
GT_Fly	0.19	51.18	0.28	40.71	0.31	51.98
Poznan_Hall2	0.38	41.77	0.31	36.24	0.32	49.33
Poznan_street	0.21	38.35	0.16	39.83	0.18	47.81
Undo_dancer	0.29	37.49	0.37	35.69	0.29	46.74
Shark	0.16	36.57	0.17	37.02	0.14	39.27
1024×768	0.2	26.92	0.27	32.99	0.24	39.6
1920×1088	0.25	41.07	0.26	37.9	0.25	47.03
Average	0.23	35.76	0.26	36.06	0.25	44.24

Observing Table 3 reveals that the amalgamation of the GLCM-based edge complexity detection algorithm with the VTM10.0 anchoring approach yields an average coding time reduction of 35.76%; the BDBR experiences a minor increase of only 0.23%, which indicates that it can effectively skip the CU partition of the flat region, and pay more attention to the CU partition of the complex and edge blocks, which underscores the efficacy of the proposed algorithm. The extra trees-based CU fast decision algorithm combined with the VTM10.0 anchoring algorithm saves 36.06% coding time on average. In comparison, the BDBR increases by only 0.26%, which indicates that the algorithm's decision on the CU partitioning method effectively reduces the coding of the depth map of the VVC 3D video. The 3D video sequences we tested have two resolutions, 1024 × 768 and 1920 × 1088. The GLCM-based edge complexity detection algorithm has the best performance in 1920 × 1088 video sequences, with an average coding time saving of up to 41.07% and a coding time saving of up to 51.18% in GT_Fly sequences. The Extra trees-based CU fast decision algorithm performs excellently in 1024 × 768 video sequences with an average coding time saving of up to 32.99% compared to the GLCM-based edge complexity detection algorithm and up to 40.71% in GT_Fly video sequences. Both sub-algorithms effectively reduce the RD cost computation and diminish the computational intricacy of VVC 3D video depth map coding.

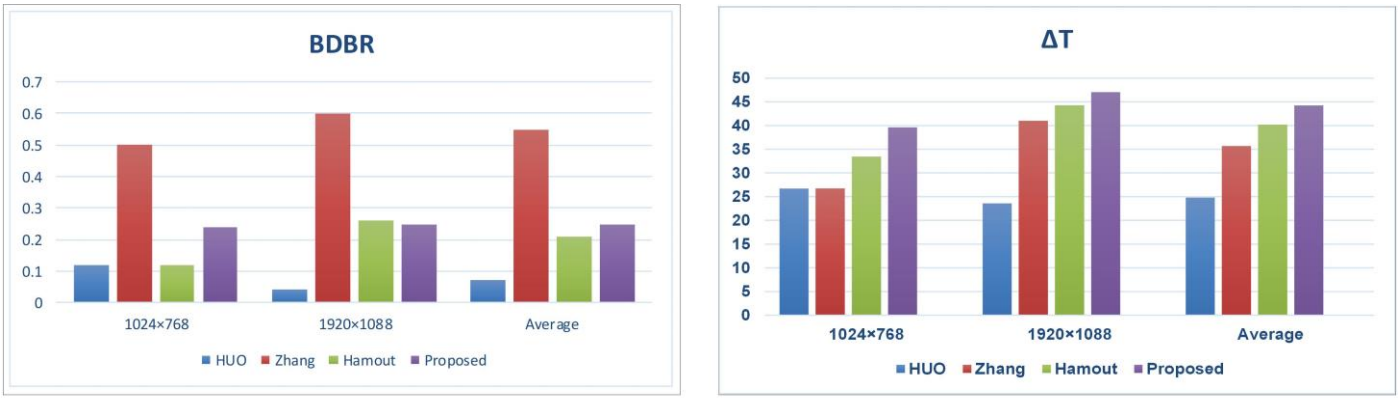
Furthermore, the coding performance of the comprehensive proposed algorithm is demonstrated in Table 3. The proposed algorithm comprises two sub-algorithms, the GLCM based edge complexity detection algorithm and the Extra trees-based CU fast decision algorithm, which can effectively diminish the coding time by an average of 44.24%. At the same time, the BDBR increases by only 0.25% simultaneously, which is negligible when contrasted with the VTM anchor algorithm. The average coding time saving is the most in the 1920 × 1088 resolution 3D video sequence with 47.03%, where the GT_Fly sequence has more flat regions; for flat blocks, the proposed sub-algorithm performs a skip operation and more flat blocks are skipped in this sequence, so the average coding time saving is the most with 51.98%. In the 1024 × 768 resolution video sequences, the newspaper sequence saves the most coding time at 43.09% (the increase in BDBR is negligible).

4.2. Comparison with Other Algorithms

To more effectively show the superiority of our algorithm in diminishing coding complexity, we chose three other excellent algorithms for performance comparison, namely, the fast rate-distortion optimization algorithm proposed by Huo et al. [39] for depth map, the convolutional neural network-based adaptive CU size coding algorithm within the frame of depth maps proposed by Zhang et al. [40], and the CU size decision algorithm employed to mitigate the intricacy of the coding depth map within the frame proposed by Hamout et al. [41]. Table 4 shows the experimental results for each algorithm and indicates that [39] achieves an average encoding time reduction of 24.8%. At the same time, the proposed algorithm surpasses [39] by achieving even more significant time savings, especially in the GT_Fly sequence, which saves the most increase in coding time; in addition, BDBR has increased but only by 0.17%, which is negligible. Then, the algorithm of [40] saves an average of 35.7% of ΔT and the proposed algorithm outperforms the comparison by saving an additional 8.54% in ΔT and the proposed algorithm has a comparatively lower BDBR which highlights the superiority of the proposed algorithm. In contrast to the algorithm of [41], the algorithm has improved by 4.04% in saving encoding time. Figures 7 and 8 illustrate the impact of the proposed algorithm on ΔT and BDBR across video sequences of varying resolutions when compared to the mentioned algorithms. It is evident that the present algorithm performs well in terms of coding performance in each resolution category. Consequently, it can be deduced that the proposed algorithm showcases significantly superior performance compared to the algorithms above. Moreover, its effectiveness in alleviating the burden of depth map coding is even more pronounced.

Table 4. Performance comparison between the proposed algorithm and the algorithms of [39–41].

Sequence	Huo [39]		Zhang [40]		Hamout [41]		Proposed	
	BDBR (%)	ΔT (%)	BDBR (%)	ΔT (%)	BDBR (%)	ΔT (%)	BDBR (%)	ΔT (%)
Balloons	0.14	27.9	0.30	22.3	0.12	32.9	0.18	39.45
Kendo	0.11	28.2	0.50	32.4	0.17	35.2	0.16	36.27
Newspaper	0.10	24.1	0.70	25.6	0.08	32.3	0.39	43.09
GT_Fly	0.05	23.5	0.80	51.7	0.08	35.0	0.31	51.98
Poznan_Hall2	0.06	27.6	0.40	42.7	0.39	51.6	0.32	49.33
Poznan_street	0.05	23.1	0.50	38.4	0.26	41.6	0.18	47.81
Undo_dancer	0.01	20.3	1.00	36.4	0.29	49.3	0.29	46.74
Shark	0.03	23.4	0.30	36.0	0.26	44.0	0.14	39.27
Average	0.07	24.8	0.55	35.7	0.21	40.2	0.24	44.24



(a) (b)

Figure 7. Comparison of the average coding performance of the proposed algorithm with that of [39–41] across video sequences of various resolutions; (**a**,**b**).

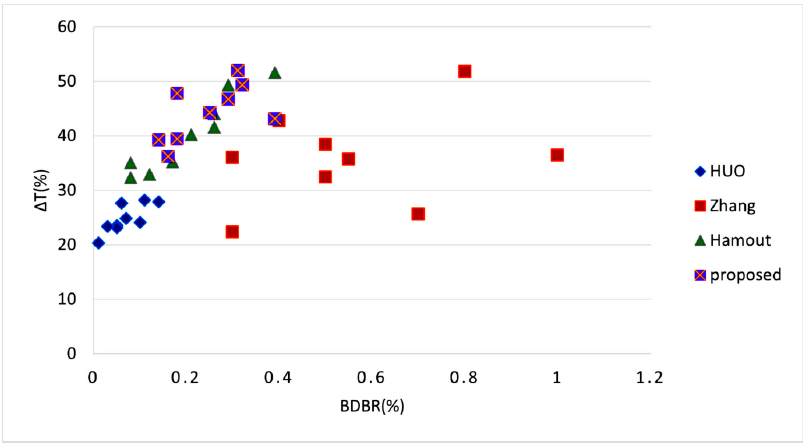


Figure 8. Comparison of ΔT and BDBR of the above algorithms.

4.3. Additional Analysis

In order to further illustrate the efficiency of the proposed algorithm in saving coding complexity, the running time consumption of the model was further analyzed by comparing the proposed method with VTM10.0. As shown in Figure 9, compared with VTM10.0, the running time overhead of the proposed algorithm in all video sequences is less than 10%, and the average time overhead is 7.89%, accounting for only a tiny part of the total encoding time. The critical reason for this result is that in the GLCM algorithm, skipping

the smooth block only further divides the edge complex blocks, and the Extra trees model makes an early decision on the CU division method, skipping most of the redundant RDO process. In addition, we also compared the details of the partition results of the sequence Poznan_street made by VTM10.0 with the partition results of the proposed algorithm, as shown in Figure 10.

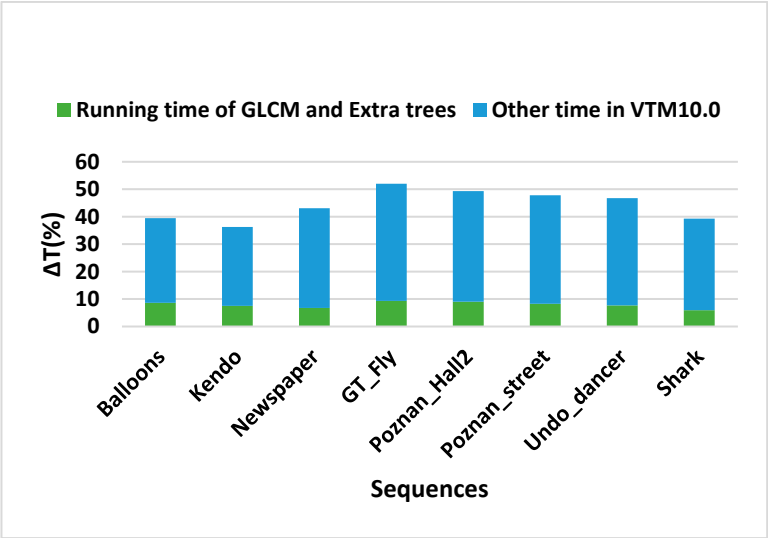


Figure 9. Runtime of the proposed algorithm and VTM encoder.

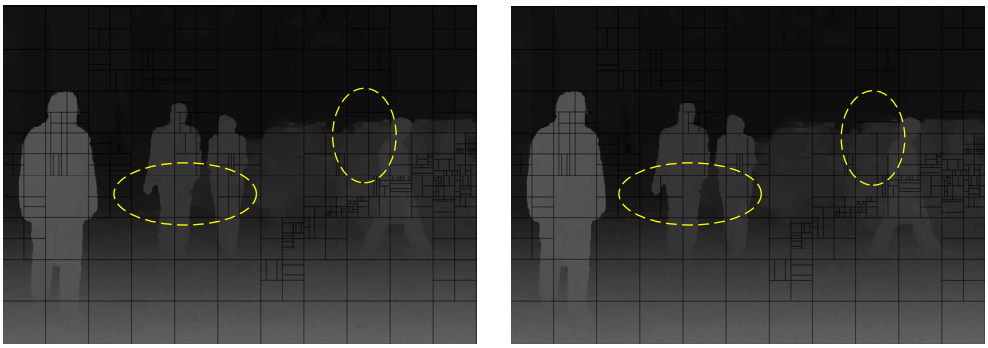


Figure 10. Example of comparison of CU partition results; (**a**) VTM10.0, (**b**) proposed algorithm. The yellow circle shows the CU division results of the same part.

5. Conclusions

To effectively address the inherent computational intricacy in depth map coding within the VVC 3D video, an algorithm is proposed with characteristics of the depth map. First, it is proposed to classify the CU using the GLCM-based edge complexity detection algorithm and perform a skip operation for the CU of the flat blocks to minimize many unnecessary RDO processes. On this basis, we propose to use the CU fast decision algorithm based on Extra trees for the CU partition method, which can judge the CU partition method in advance, further avoid unnecessary RD cost calculations, and further enhance the reduction of coding complexity. The research demonstrated that the proposed algorithm substantially diminishes the encoding burden, achieves an average reduction of 44.24% in coding time, and increases BDBR by only 0.25% (negligible), which showcases the exceptional performance of the proposed algorithm. Furthermore, we will further improve the Extra trees model so that the CU partition method can be accurately judged and also test the video sequences with more resolutions to expand the scope of the application.

Author Contributions: Conceptualization, F.W. and Z.W.; methodology, F.W.; software, Z.W.; validation, F.W., Q.Z. and Z.W.; formal analysis, Z.W.; investigation, Z.W.; resources, Q.Z.; data curation, Z.W.; writing—original draft, Z.W.; writing—review and editing, F.W.; visualization, F.W.; 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, and the Key projects Natural Science Foundation of Henan 232300421150, the Scientic and Technological Project of Henan Province 232102211014, and the Postgraduate Education Reform and Quality Improvement Project of Henan Province YJS2021KC12, YJS2023JC08, and YJS2022AL034.

Data Availability Statement: Not applicable.

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

Abbreviations

The following abbreviations are used in this manuscript:

VVC	Versatile Video Coding
QTMT	Quad-tree with Nested Multi-type Tree
CU	Coding Unit
GLCM	Gray Level Co-occurrence Matrix
BDBR	Bjøntegaard Delta Bit Rate
HDR	High Dynamic Range
VR	Virtual Reality
AR	Augmented Reality
MPEG	Moving Picture Experts Group
VCEG	Video Coding Expert Group
HEVC	High Efficiency Video Coding
JVET	Joint Video Experts Team
RDO	Rate Distortion Optimization
DMM	Depth Modeling Modes
CNN	Convolutional Neural Network
FNN	Feedforward neural network
VGGNet	Visual Geometry Group Network
QT	Quad Tree
BTH	Horizontal Binary Tree
BTV	Vertical Binary Tree
TTH	Horizontal Trinomial Tree
TTV	Vertical Trinomial Tree
ASM	Angle-Second Matrix
CON	Contrast
COR	Correlation
MTT	Multi-Type Tree
QP	Quantization Parameter
VTM	VVC Test Model

References

1. Cheon, M.; Lee, J.-S. Subjective and Objective Quality Assessment of Compressed 4K UHD Videos for Immersive Experience. *IEEE Trans. Circuits Syst. Video Technol.* **2018**, *28*, 1467–1480. [CrossRef]
2. Muller, K.; Merkle, P.; Wiegand, T. 3-D Video Representation Using Depth Maps. *Proc. IEEE* **2011**, *99*, 643–656. [CrossRef]
3. Boyce, J.M.; Dore, R.; Dziembowski, A.; Fleureau, J.; Jung, J.; Kroon, B.; Salahieh, B.; Vadakital, V.K.M.; Yu, L. MPEG Immersive Video Coding Standard. *Proc. IEEE* **2021**, *109*, 1521–1536. [CrossRef]
4. Aggoun, A.; Tsekleves, E.; Swash, M.R.; Zarpalas, D.; Dimou, A.; Daras, P.; Nunes, P.; Soares, L.D. Immersive 3D Holoscopic Video System. *IEEE MultiMed.* **2013**, *20*, 28–37. [CrossRef]
5. Chen, Y.; Vetro, A. Next-Generation 3D Formats with Depth Map Support. *IEEE MultiMed.* **2014**, *21*, 90–94. [CrossRef]

6. Lei, J.; Shi, Y.; Pan, Z.; Liu, D.; Jin, D.; Chen, Y.; Ling, N. Deep Multi-Domain Prediction for 3D Video Coding. *IEEE Trans. Broadcast.* **2021**, *67*, 813–823. [CrossRef]
7. Liu, C.; Jia, K.; Liu, P. Fast Depth Intra Coding Based on Depth Edge Classification Network in 3D-HEVC. *IEEE Trans. Broadcast.* **2022**, *68*, 97–109. [CrossRef]
8. Tech, G.; Chen, Y.; Muller, K.; Ohm, J.-R.; Vetro, A.; Wang, Y.-K. Overview of the Multiview and 3D Extensions of High Efficiency Video Coding. *IEEE Trans. Circuits Syst. Video Technol.* **2016**, *26*, 35–49. [CrossRef]
9. Tissier, A.; Mercat, A.; Amestoy, T.; Hamidouche, W.; Vanne, J.; Menard, D. Complexity reduction opportunities in the future VVC intra encoder. In Proceedings of the 21th International Workshop Multimedia Signal Process (MMSP), Kuala Lumpur, Malaysia, 27–29 September 2019; pp. 1–6.
10. 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.
11. Park, C.-S. Edge-Based Intramode Selection for Depth-Map Coding in 3D-HEVC. *IEEE Trans. Image Process.* **2015**, *24*, 155–162. [CrossRef]
12. Sanchez, G.; Silveira, J.; Agostini, L.V.; Marcon, C. Performance Analysis of Depth Intra-Coding in 3D-HEVC. *IEEE Trans. Circuits Syst. Video Technol.* **2019**, *29*, 2509–2520. [CrossRef]
13. Zhang, Y.; Lu, C. Efficient algorithm adaptations and fully parallel hardware architecture of H. 265/HEVC intra encoder. *IEEE Trans. Circuits Syst. Video Technol.* **2018**, *29*, 3415–3429. [CrossRef]
14. 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]
15. Li, H.; Zhang, P.; Jin, B.; Zhang, Q. Fast CU Decision Algorithm Based on CNN and Decision Trees for VVC. *Electronics* **2023**, *12*, 3053. [CrossRef]
16. Li, T.; Wang, H.; Chen, Y.; Yu, L. Fast depth intra coding based on spatial correlation and rate distortion cost in 3D-HEVC. *Signal Process. Image Commun.* **2020**, *80*, 115668. [CrossRef]
17. Zuo, J.; Chen, J.; Zeng, H.; Cai, C.; Ma, K.-K. Bi-Layer Texture Discriminant Fast Depth Intra Coding for 3D-HEVC. *IEEE Access.* **2019**, *7*, 34265–34274. [CrossRef]
18. Fu, C.-H.; Chan, Y.-L.; Zhang, H.-B.; Tsang, S.H.; Xu, M.-T. Efficient Depth Intra Frame Coding in 3D-HEVC by Corner Points. *IEEE Trans. Image Process.* **2021**, *30*, 1608–1622. [CrossRef]
19. Li, T.; Yu, L.; Wang, H.; Chen, Y. Fast depth intra coding based on texture feature and spatio-temporal correlation in 3D-HEVC. *IET Image Process.* **2021**, *15*, 206–217. [CrossRef]
20. Hamout, H.; Elyousfi, A. An efficient edge detection algorithm for fast intra-coding for 3D video extension of HEVC. *J. Real-Time Image Proc.* **2019**, *16*, 2093–2105. [CrossRef]
21. Li, Y.; Yang, G.; Qu, A.; Zhu, Y. Tunable early CU size decision for depth map intra coding in 3D-HEVC using unsupervised learning. *Digit. Signal Process.* **2022**, *123*, 103448. [CrossRef]
22. Zhang, Z.; Yu, L.; Qian, J.; Wang, H. Learning-Based Fast Depth Inter Coding for 3D-HEVC via XGBoost. In Proceedings of the 2022 Data Compression Conference (DCC), Snowbird, UT, USA, 22–25 March 2022; pp. 43–52.
23. Saldanha, M.; Sanchez, G.; Marcon, C.; Agostini, L. Fast 3D-HEVC Depth Map Encoding Using Machine Learning. *IEEE Trans. Circuits Syst. Video Technol.* **2020**, *30*, 850–861. [CrossRef]
24. Zhang, R.; Jia, K.; Liu, P. Fast CU Size Decision Using Machine Learning for Depth Map Coding in 3D-HEVC. In Proceedings of the 2020 Data Compression Conference (DCC), Snowbird, UT, USA, 24–27 March 2020; p. 405.
25. Fu, C.-H.; Chen, H.; Chan, Y.-L.; Tsang, S.-H.; Hong, H.; Zhu, X. Fast Depth Intra Coding Based on Decision Tree in 3D-HEVC. *IEEE Access* **2019**, *7*, 173138–173147. [CrossRef]
26. Peng, B.; Chang, R.; Pan, Z.; Li, G.; Ling, N.; Lei, J. Deep In-Loop Filtering via Multi-Domain Correlation Learning and Partition Constraint for Multiview Video Coding. *IEEE Trans. Circuits Syst. Video Technol.* **2023**, *33*, 1911–1921. [CrossRef]
27. Liu, C.; Jia, K.; Liu, P.; Sun, Z. Fast Depth Intra Coding Based on Layer-Classification and CNN for 3D-HEVC. In Proceedings of the 2020 Data Compression Conference (DCC), Snowbird, UT, USA, 24–27 March 2020; p. 381.
28. Zhang, J.; Hou, Y.; Zhang, Z.; Jin, D.; Zhang, P.; Li, G. Deep region segmentation-based intra prediction for depth video coding. *Multimed. Tools Appl.* **2022**, *81*, 35953–35964. [CrossRef]
29. Xie, S.; Tu, Z. Holistically-Nested Edge Detection. *Int. J. Comput. Vis.* **2017**, *125*, 3–18. [CrossRef]
30. Guo, L.; Tian, X.; Chen, Y. Simplified depth intra coding for 3D-HEVC based on gray-level co-occurrence matrix. In Proceedings of the 2016 IEEE International Conference on Signal and Image Processing (ICSIP), Beijing, China, 13–15 August 2016; pp. 328–332.
31. Chen, J.; Liao, J.; Zuo, J.; Zeng, H.; Cai, C.; Ma, K.-K. Fast Depth Intra-Coding for 3D-HEVC based on Gray-Level Co-occurrence Matrix. *J. Imaging Sci. Technol.* **2019**, *63*, 30406. [CrossRef]
32. Haralick, R.M.; Shanmugam, K.; Dinstein, I. Textural Features for Image Classification. *IEEE Trans. Syst.* **1973**, *6*, 610–621.
33. Chen, J.; Sun, H.; Katto, J.; Zeng, X.; Fan, Y. Fast QTMT Partition Decision Algorithm in VVC Intra Coding based on Variance and Gradient. In Proceedings of the 2019 IEEE Visual Communications and Image Processing (VCIP), Sydney, Australia, 1–4 December 2019; pp. 1–4.
34. Qian, X.; Zeng, Y.; Wang, W.; Zhang, Q. Co-Saliency Detection Guided by Group Weakly Supervised Learning. *IEEE Trans. Multimed.* **2023**, *25*, 1810–1818. [CrossRef]

35. Otroshi-Shahreza, H.; Amini, A.; Behroozi, H. Feature-based no-reference video quality assessment using Extra Trees. *IET Image Process.* **2022**, *16*, 1531–1543. [CrossRef]
36. Park, S.; Kang, J.-W. Fast Multi-Type Tree Partitioning for Versatile Video Coding Using a Lightweight Neural Network. *IEEE Trans. Multimed.* **2021**, *23*, 4388–4399. [CrossRef]
37. Li, Q.; Meng, H.; Li, Y. Texture-based fast QTMT partition algorithm in VVC intra coding. *Signal Image Video Process.* **2023**, *17*, 1581–1589. [CrossRef]
38. Bjontegaard, G. Calculation of Average PSNR Differences Between RD Curves. In Proceedings of the ITU SG16 Doc. VCEG-M33, Austin, TX, USA, 2–4 April 2001.
39. Huo, J.; Zhou, X.; Yuan, H.; Wan, S.; Yang, F. Fast Rate-Distortion Optimization for Depth Maps in 3-D Video Coding. *IEEE Trans. Broadcast.* **2023**, *69*, 21–32. [CrossRef]
40. Zhang, H.; Yao, W.; Huang, H.; Wu, Y.; Dai, G. Adaptive coding unit size convolutional neural network for fast 3D-HEVC depth map intracoding. *J. Electron. Imag.* **2021**, *30*, 4. [CrossRef]
41. Hamout, H.; Elyousfi, A. A Computation Complexity Reduction of the Size Decision Algorithm in 3D-HEVC Depth Map Intracoding. *Adv. Multimed.* **2022**, *2022*, 3507201. [CrossRef]