# Efficient Intra Prediction in Versatile Video Coding Using Adaptive Partitioning and Structural Analysis

Original Scientific Paper

**Naga Nrutya Praveena Veerapaneni**
VIT-AP University,
School of Electronics Engineering, Inavolu, Beside AP Secretariat, Amaravati,
Andhra Pradesh, 522237, India.
veerapaneni.praveena@gmail.com

**Purnachand Nalluri***
VIT-AP University,
School of Electronics Engineering, Inavolu, Beside AP Secretariat, Amaravati,
Andhra Pradesh, 522237, India.
chanduinece@gmail.com

*Corresponding author

***Abstract*** *– Versatile Video Coding (VVC) implements numerous advanced tools for improving compression efficiency and perceptual video quality at the expense of a substantial increase in computing complexity. One such tool is intra-frame coding. To resolve this computational problem, this paper proposes an efficient intra-coding approach using texture and structural analysis. The initial texture characteristics are derived through the early termination of coding unit (CU) splitting using the Scharr operator, which captures directional gradient information. Next, the extraction of structural features is based on a dissimilarity-driven criterion to skip redundant ternary tree (TT) splits and thus facilitate early termination of ternary CU partitioning. Compared with state-of-the-art methods, the proposed method yields a 45.79% decrease in encoding time with a minimal increase of 1.23% in BD-bitrate across standard test sequences.*

***Keywords****:* *intra prediction, scharr operator, versatile video coding, video compression*

*Received: July 25, 2025; Received in revised form: November 22, 2025; Accepted: December 4, 2025*

## 1. INTRODUCTION

The rapid growth of multimedia technologies and 5G has encouraged significantly greater demand for high-resolution content, such as 4K/8K videos and virtual reality videos [1]. This demand has already outgrown the efficiency of conventional codecs, including Advanced Video Coding (H.264/AVC) and High Efficiency Video Coding (H.265/HEVC), and requires more effective video compression solutions [2]. The new video coding standard VVC, released in July 2020, is more flexible and efficient compared to HEVC [3]. In VVC, a coding unit and coding tree unit are partitioned by using the QTMT (Quadtree with Nested Multi-Type Tree) structure of a versatile quadtree. The QTMT structure utilizes five distinct split patterns: binary vertical and horizontal, ternary vertical and horizontal, and quaternary. This enables better video compression and stronger adaptability to diverse content types and scenes than ever [4]. The MTT structure has used four kinds of splits: ternary vertical (TV), ternary horizontal (TH), binary vertical (BV), and binary horizontal (BH) [5]. In this system, the binary splits equally divide the coding blocks into two halves, while the ternary splits divide them into three segments of a 1:2:1 ratio. With the QTMT structure of this VVC, the flexibility enables the coding block to vary in shape and size according to the texture and content features of the video that is being encoded.

The QTMT partitioning model in VVC is designed to ensure the size and shape of the coding blocks based on the texture characteristics of the content. A VVC luma CTU supports a luma coding tree unit having a largest possible dimension of $128 \times 128$ pixels. By default, the CTU acts as the root node of a quadtree, from which CUs are derived through quadtree-based partitioning. The VVC test model (VTM) determines the rate-distortion (RD) cost for each possible partitioning mode and chooses the one with the lowest possible value in the rate-distortion optimization (RDO) procedure. As shown in Fig. 1, the evaluation order is no split (NS), quad-tree (QT), binary horizontal (BH), binary vertical (BV), ternary horizontal (TH), and ternary vertical (TV). The cost of RD is calculated using a designated formula.

$$J = D + \lambda B \tag{1}$$

In equation (1), $D$ represents the sum of absolute differences between the original and reconstructed coding units, $B$ is the bit count for the presently selected mode, and $\lambda$ denotes the Lagrangian multiplier.

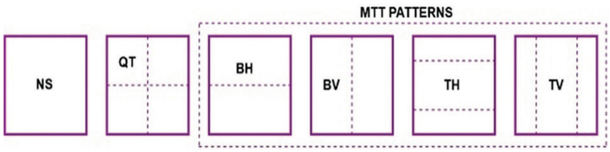



**Fig. 1.** Evaluation order of QTMT Structure

The QTMT structure in VVC improves intra coding performance but adds computational complexity. VVC intra-coding is significantly slower than HEVC, and its encoding duration is about 31 times longer, but the disabled multi-type tree (MTT) partitioning can reduce the encoding time by 83% [6]. This high complexity hampers VVC's practical use. Since many CUs have recognizable texture features and partition trends, predicting these patterns early could reduce the encoding time without affecting quality. We present a QTMT partitioning approach that uses the Scharr operator and subblock edge comparison to compute gradients. These gradients classify coding units into simple or complex textures. For simple textures, partitioning terminates early, while complex textures follow our proposed partitioning method. Gradient information thus predicts vertical and horizontal splits, while subblock comparisons determine if the ternary tree partitioning can be avoided. The proposed algorithm achieves a 45.79% time savings with a mere increase of 1.23% regarding Bjontegaard Delta Bitrate (BDBR). Compared to other state-of-the-art fast CU partitioning algorithms, the contributions of the proposed method can be summarized as follows:

1. The Scharr operator is used to compute the gradient that will allow for fast decision-making about partitioning and can prevent a partition from splitting vertically or horizontally.
2. We utilize the dissimilarity feature to decide whether to skip the ternary tree structure or to apply the specific partition. As our proposed method uses only two features, it is computationally efficient.
3. Traditional deep learning and machine learning methods require large training datasets. These models usually have limited adaptability and may need to be retrained separately for coding units of different sizes.

This paper is structured in five main sections. Section 2 provides a summary of previous research on reducing VVC intra prediction complexity. Section 3 explains the suggested fast QTMT partitioning technique. Section 4 discusses the experimental outcomes, and Section 5 concludes the paper with final remarks.

## 2. RELATED WORKS

Multiple approaches have recently been introduced to decrease the amount of computation associated with decisions on the size of VVC encoder CUs. These methods seek to maintain high compression performance while maintaining the trade-off between computational cost and encoding efficiency. In [7], a block-level Canny edge detector was utilized to derive the edge characteristics, allowing the exclusion of specific vertical or horizontal partition modes. An efficient intra partitioning algorithm was introduced in [8] for VVC using variance of sub-CUs and Sobel-based gradient features, where smooth areas are early terminated from further splitting. An early termination technique for CU partitioning, based on four directional gradients at 0, 45, 90, and 135 degrees, was proposed in [9]. An advanced fast intra mode selection technique for VVC was presented in [10] that employs two strategies: sorting modes according to costs of the rough mode decision (RMD) method and decreasing candidate modes by examining correlations with neighboring blocks. Based on this sorted order, an early termination technique is used to stop the prediction mode decision-making process, therefore lowering coding complexity without increasing processing cost. In [11], the utilization of the gradient computed by the Sobel operator serves as a criterion to determine whether to initiate QT partitioning. If division is considered unnecessary, subsequent sub-CU variance calculations are performed. A partition is selected from a pool of five candidate QTMT partitions. This optimization approach produced a statistically significant reduction in encoding time. An efficient early intra-CU partitioning technique for VVC is presented in [12], where the distortion is calculated as the difference between the original and predicted luminance pixels. Two decision models for premature termination of multi-type tree partitions are developed using this distortion, which minimizes insignificant calculations. Then, depending on particular needs, a tunable decision model is used, providing flexibility to determine a compromise between coding efficiency and computational complexity. A computationally efficient intra coding algorithm with two main components: fast CU partitioning decision and fast intra mode selection was introduced by [13]. The CU partition decision employs a Bayesian-based method to bypass vertical splits and enhance accuracy by merging multiple feature eigenvectors. The Limit-Broyden-Fletcher-Goldfarb-Shanno(L-BFGS) approach is used in the intra mode selection to decrease the Rough Mode Decision (RMD) and Rate Distortion Optimization (RDO) computations. An optimized CU partitioning technique is presented in [14], which uses the sum of mean absolute deviation (SMAD) to quantify and compare horizontal and vertical texture complexities, therefore reducing unlikely partition modes. The quantization parameter (QP) will be used to dynamically adjust the thresholds of directional SMAD ratios for better prediction. In [15], an effective CU partitioning technique for VVC intra prediction leverages texture and neighboring partition information to reduce computational burden, resulting in negligible degradation in coding performance. The method explores correlations between spatial texture characteristics and CU split decisions, designing decision algorithms for various sizes of CU while integrating neighboring CU division information. In [16], with the use of the Laplacian of Gaussian (LOG) operator, a low-complexity CU partition algorithm efficiently reduced the encoding time by skipping unnecessary partitions based on edge features. Overall, these methods achieve significant reductions in encoding time with reasonable coding efficiency. Although many fast decision methods have been proposed, most of them still employ fixed thresholds and handcrafted features concerning texture variance, edge strength, gradient magnitude, and directional complexity. While those features are powerful in capturing the local structure, their predefined nature leads to limited adaptability for diverse video content and different sizes of CUs.

In [17], a Support Vector Machine (SVM)-based algorithm was introduced to predict CU split decisions in VVC by analyzing features like entropy, texture contrast, and Haar wavelet coefficients. Online training of six SVM models across varying CU sizes is used to effectively forecast splitting directions. By utilizing texture information, [18] introduced a fast CU splitting technique derived from SVM classifiers that avoided replication of partitioning modes. They used quantization parameters, variance, and directional gradients of current CU as three features to train the classifiers for various CU sizes independently. With a view to decrease the complexity of the CU division, [19] proposed a random forest-based technique in VVC by categorizing the CU as simple, fuzzy or complex. A deep learning approach was proposed by [20] to estimate CU partitioning based on QTMT for advanced intra-mode selection in VVC encoding. CU partition decisions are guided by a multi-stage exit CNN (MSE-CNN) that optimizes among complexity and RD performance. It includes an early-exit mechanism and an adaptive loss function. The experimental results show encoding time reductions of 44.65% to 66.88%, with a minimal increase in BDBR of 1.322% to 3.188%, compared to VTM-7.0. A hybrid framework was proposed in [21] that combined multi-feature-guided Fast CU Partitioning (FCP) and Laplacian-guided Fast Mode Selection (FMS) to speed up intra QTMT decisions. FCP relies on SVM-based classification, while FMS employs the Laplace operator to eliminate redundant mode evaluations. Together, the methods achieved a reduction in encoding time of up to 54.84%, with only a 1.74% increase in BDBR.

A convolutional neural network (CNN) was used to describe the Quadtree plus Binary Tree (QTBT) splitting depth as a 5-class classification issue in [22], which presented a fast CU depth selection method. The CNN greatly accelerates the decision-making process by directly predicting the depth range instead of evaluating splits at each level. In [23], a CNN that works well with coding units of different sizes was trained. In CNN, the dimension of the pooling layer was linked to the size of the CU, thereby resolving the drawback of the prior network's inability to simultaneously accept CUs of varying sizes. In [24], an adaptive coding unit split decision method for VVC is proposed, which uses deep learning and multi-feature fusion. The texture classification identifies complex and homogeneous CUs, while CNN structures are applied based on CU characteristics. For square CUs, [25] introduced a CNN model to enhance the QT split decision. The model concentrates on coding units that have depths between 0 and 2. Their technique can improve the bit rate by 1.7% while decreasing the intra-mode processing time by approximately 35%. A DenseNet-based VVC fast CU partition decision scheme was implemented by [26]. The probability vector is utilized to prevent unimportant RDO procedures before they occur, and CNN is employed to estimate the possibility that the boundary of $4 \times 4$ blocks in each $64 \times 64$ block will serve as the partition boundaries. The use of CNNs in a deep learning method was suggested in [27] as a fast intra-partitioning solution for VVC. A novel and efficient QTMT decision tree technique is created by combining the predictions of a CNN-based binary tree horizontal (CNN-BTH) and binary tree vertical (CNN-BTV) network for partition choices at the $32 \times 32$ CU level. The suggested approach reduces complexity by 37%, saves 31% of the encoding time, and only slightly degrades coding performance when compared to VTM-3.0. For intra-frame analysis within the VVC standard, a fast method for selecting block divisions using LGBM (Light Gradient Boosting Machine) classifiers was presented in [28] to determine optimal split types among the five possible options in the QTMT structure of VVC. This approach uses five specialized classifiers, trained for texture, coding, and context features, to predict and skip unlikely split types, which avoids unnecessary RDO calculations. This configurable solution offers multiple operation points that balance performance and efficiency. A two-stage framework was suggested in [29] to make VVC less complicated when predicting frames. To identify the optimal intra-coding depth, a deep convolutional network first captures and combines spatial-temporal coding characteristics. Next, a probability-based model selects possible partition modes, reducing unnecessary calculations. In [30], a learning-based approach was proposed for fast split mode and directional mode decisions in VVC intra prediction using deep neural networks. Separate models were trained for different CU sizes to predict the probabilities of split modes and directional modes, enabling the skipping of unlikely candidates and early termination using a lightweight neural network. In [31], a dynamic decision-making mechanism for fast CU partitioning using the ResNet model with a gradient-based intra-mode selection was implemented. Although learning-based methods are more accurate, they require multiple models to be trained on large datasets with specialized architectures, which significantly increases the computational overhead at inference.

From the analysis of existing methods, it is evident that achieving a balance between adaptability and efficiency remains challenging. To address this, the proposed method performs a perceptually guided CU partitioning using the Scharr operator for the detection of edge direction and an SSIM-based dissimilarity (Dis) feature for adaptive partitioning. This simple approach achieves significant complexity reduction with minimal loss in coding efficiency, without relying on fixed thresholds or learning model.

## 3. PROPOSED METHOD

VVC is based on a QTMT framework that offers superior performance in video coding while entailing much longer encoding times. To reduce complexity, we have proposed a fast CU size decision using the Scharr operator and the subblock dissimilarity measure. In the proposed system, we use the default VTM settings, which are 64 for MaxCUWidth and MaxCUHeight, and thus, a QT split is required for 128 × 128 CTUs. For VVC's All Intra (AI) setup, the largest base node size that can be supported by the multi-type tree structure is 32 × 32. This limitation is because MAX_BT_SIZE and MAX_TT_SIZE are set to 32. Consequently, for 64 × 64 CUs, there is a preference between splitting and not splitting. We propose early splitting decisions for CUs between 16 × 16 and 32 × 32, which achieve a tradeoff between efficiency and coding quality, considering that the early termination of 64 × 64 CUs may have a limited loss of coding quality.

### 3.1. MOTIVATION

For the QTMT coding structure of VVC, intra-coding consumes more than 90% of the coding time [13]. However, using QTMT to explore the best partitioning pattern includes redundant computation of tree traversals since all the potential partitioning patterns perform RD testing. By identifying video features to prevent splitting into inappropriate types, we can significantly reduce the VVC encoding complexity. The QTMT partitioning is closely related to the contents textures and tends to choose larger CU sizes for homogeneous areas with a simple coding tree structure, while areas with complex textures tend to have smaller CU sizes and complex coding tree structures. For areas with oriented textures, CUs are partitioned to keep the texture direction aligned for optimal encoding.

### 3.2. OVERVIEW OF FEATURE EXTRACTION

The two categories that the video features can be divided into include texture and structural features. While structural features capture the arrangement of subblocks within the encoding block, texture features describe the correlation between the coding unit partitioning and pixel variations. Employment of gradient information helps in describing the texture properties of the CU accurately. The Scharr operator has been adopted for gradient extraction to efficiently enhance the accuracy of the gradient. Equations (2), (3), (4) and (5) illustrate how convolution factors are calculated horizontally and vertically and gradients in the Scharr operator.

$$G_h(p,q) = \begin{pmatrix} -3 & 0 & 3 \\ -10 & 0 & 10 \\ -3 & 0 & 3 \end{pmatrix} * A(p,q) \tag{2}$$

$$G_v(p,q) = \begin{pmatrix} -3 & -10 & -3 \\ 0 & 0 & 0 \\ 3 & 10 & 3 \end{pmatrix} * A(p,q) \tag{3}$$

$$G_h = \frac{1}{W \times H} \sum_{p=0}^{W-1} \sum_{q=0}^{H-1} |G_h(p,q)| \tag{4}$$

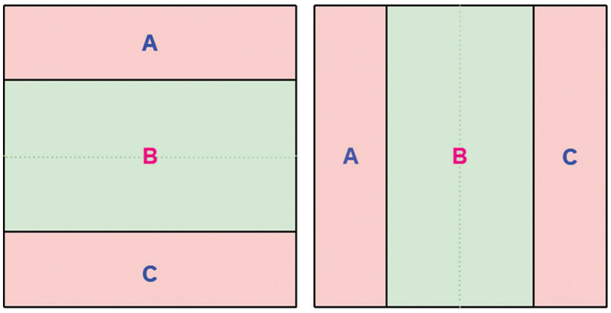
$$G_v = \frac{1}{W \times H} \sum_{p=0}^{W-1} \sum_{q=0}^{H-1} |G_v(p,q)| \tag{5}$$

A($p$, $q$) represents the matrix of pixels centered on ($p$, $q$), while $G_h$ ($p$, $q$) and $G_v$ ($p$, $q$) represent the values of the horizontal and vertical gradients at the coordinates ($p$, $q$). $G_h$ and $G_v$ represent the aggregate horizontal and vertical gradients. The current area is divided into four equally sized subblocks horizontally and vertically. The width and height are given by $W$ $an$d H, respectively, and the average of the absolute gradient values for every single pixel is calculated.

The BT (Binary Tree) and TT (Ternary Tree) partitions divide the CU autonomously into subblocks, showing its structure in detail. The CU must be partitioned uniformly into four parts both horizontally and vertically to compare it fairly. This equal-sized representation is schematic and is used here only for visualization; in actual VVC, BT and TT partitions produce non-uniform subblock dimensions. Fig. 2 presents the positions of the subblocks generated by the partition and helps us to provide an insight into how subblock partitioning is performed. In the partitioning in the horizontal direction, subblocks A and C are the edge subblocks in the TT partition, while subblock B is the middle subblock. In this way, it is possible to investigate the similarities in the structure of the subblock.

One of the well-known image quality metrics used to evaluate image fidelity is the Structural Similarity Index Measure (SSIM) [32]. Unlike variance, SSIM considers not only contrast but also luminance and the structural correlation between the edges of subblocks. Therefore, SSIM can be applied to quantify the comparison between the edges of the adjacent subblocks. In SSIM, structural similarity is measured by considering luminance, contrast, and structure. SSIM is more perceptually accurate since it correlates more with human visual perception compared to variance. It is used to characterize the present CU's structural properties to avoid certain multi-type tree partitioning patterns from occurring early. It is particularly useful for image quality evaluation and can be applied effectively to calculate the dissimilarity feature by comparing the SSIM values of adjacent subblock edges.

**Fig. 2.** Visualization of Horizontal and Vertical Subblock Division




SSIM compares subblocks by analyzing not just pixel intensity differences (as variance does), but also structural similarities, which are more aligned with human perception.

- Luminance differences are addressed by comparing mean pixel intensities.
- Contrast differences are captured through variance, but with a more perceptually accurate formulation.
- Structural differences are captured by using covariance in SSIM, reflecting the similarity of the structures (patterns) between subblocks.

The formula represented in Equation (6) below provides the SSIM measurement between the two edge pixel arrays, $y$ and $z$.

$$SSIM(y,z) = \frac{(2\mu_y\mu_z+C_1)(2\sigma_{yz}+C_2)}{(\mu_y^2+\mu_z^2+C_1)(\sigma_y^2+\sigma_z^2+C_2)} \tag{6}$$

where

The mean values of the pixels along the edges in the subblocks $y$ and $z$ are $\mu_y$ and $\mu_z$.

Edge pixel values in the subblocks have variances of $\sigma_y^2$ and $\sigma_z^2$.

The covariance between the edge pixel values of the adjacent subblocks is $\sigma_{yz}$.

The constants $C_1$ and $C_2$ are used to stabilize divisions with weak denominators. In general, $C_1 = (K_1L)^2$ and $C_2 = (K_2L)^2$; here $L$ represents the dynamic range of the pixel values (for example, 1023 for 10-bit images and 255 for 8-bit images). In standard SSIM implementations, the default values for $K_1$ and $K_2$ are 0.01 and 0.03 respectively.

SSIM takes into account not only the contrast (variance) but also the luminance and structural correlation between the subblocks. Therefore, SSIM provides a more complete picture by evaluating:

1. Luminance similarity (mean intensity of the subblocks)
2. Contrast similarity (variance or standard deviation of the subblocks)
3. Structural similarity (covariance between subblocks)

The mean luminance in each block is calculated using the average pixel values of the edge blocks as represented in Equation (7)

$$L(y,z) = \frac{2\mu_y\mu_z+C_1}{\mu_y^2+\mu_z^2+C_1} \tag{7}$$

The mean value for each subblock is given by the following equations (8) and (9)

$$\mu_y = \frac{1}{P}\sum_{s=1}^{P} y_s \tag{8}$$

$$\mu_z = \frac{1}{P}\sum_{s=1}^{P} z_s \tag{9}$$

Here, $P$ represents the overall pixel count along the edge in the region (or block) of the image being analyzed. Contrast differences are captured through variance, but with a more perceptually accurate formulation.

$$C(y,z) = \frac{2\sigma_y\sigma_z+C_2}{\sigma_y^2+\sigma_z^2+C_2} \tag{10}$$

The variances $\sigma_y^2$ and $\sigma_z^2$ of the edge pixel values in subblocks $y$ and $z$ are given by the following equations.

$$\sigma_y^2 = \frac{1}{P-1}\sum_{s=1}^{P}(y_s - \mu_y)^2 \tag{11}$$

$$\sigma_z^2 = \frac{1}{P-1}\sum_{s=1}^{P}(z_s - \mu_z)^2 \tag{12}$$

The covariance in SSIM captures structural differences, indicating the similarity between structures (patterns) in subblocks. It can be computed using the following equation.

$$S(y,z) = \frac{\sigma_{yz}+C_2}{\sigma_y\sigma_z+C_2} \tag{13}$$

The covariance $\sigma_{yz}$ between the edge values of the pixels in the subblocks $y$ and $z$ is given by the following equation.

$$\sigma_{yz} = \frac{1}{P-1}\sum_{s=1}^{P}(y_s - \mu_y)(z_s - \mu_z) \tag{14}$$

Combine the results into the SSIM formula represented in Equation (6).

In the context of video coding and partitioning decisions, measuring dissimilarity is important because the goal is often to identify regions of the image or video frame where the structure changes significantly.

These changes can indicate the need for further partitioning to improve coding efficiency by better capturing the varying details within different blocks. When using SSIM to drive partitioning decisions, dissimilarity can guide whether to skip or apply a specific partition. In this case, the measure of dissimilarity (Dis) can be represented as:

$$
\begin{aligned}
Dis_{bh} &= 1\text{-}SSIM(BE\_Subblock\_{}_{h4}, TE\_Subblock\_{}_{h3})\\
Dis_{th1} &= 1\text{-}SSIM(BE\_Subblock\_{}_{h1}, TE\_Subblock\_{}_{h2})\\
Dis_{th2} &= 1\text{-}SSIM(BE\_Subblock\_{}_{h3}, TE\_Subblock\_{}_{h4})\\
Dis_{bv} &= 1\text{-}SSIM(RE\_Subblock\_{}_{v4}, LE\_Subblock\_{}_{v3})\\
Dis_{tv1} &= 1\text{-}SSIM(RE\_Subblock\_{}_{v1}, LE\_Subblock\_{}_{v2})\\
Dis_{tv2} &= 1\text{-}SSIM(RE\_Subblock\_{}_{v3}, LE\_Subblock\_{}_{v4})
\end{aligned}
\tag{15}
$$

In the above equation (15), the terms $BE\_Subblock$, $TE\_Subblock$, $RE\_Subblock$ and $LE\_Subblock$ refer to the bottom edge, top edge, right edge and left edge of the subblock, respectively. As illustrated in Fig. 2, the image is partitioned into four distinct subblocks. To facilitate analysis, the terms $h1$, $h2$, $h3$ and $h4$ denote horizontal edge pixels and v1, $v2$, $v3$ and $v4$ denote vertical edge pixels. During encoding, SSIM is calculated across the edges of neighboring subblocks in both the horizontal and vertical directions using reconstructed pixels. If the dissimilarity is below an adaptive threshold, the CU is homogeneous and no further splits are necessary. Otherwise, the CUs with high $Dis$ values will undergo finer partitioning to make more precise predictions. The adaptive control minimizes redundant evaluations in smooth areas while preserving detail in complex regions. This feature is implemented into the EncCu module of the VTM encoder before a full RD cost calculation.

### 3.3. PROPOSED ALGORITHM

Our model presents a unified early decision strategy for the QTMT coding structure of VVC with the goal of optimizing coding efficiency by simplifying RDO mode decisions. By focusing on reducing the number of coding units involved in RDO processing, we effectively reduce overhead in coding time. In this paper, we focus on further enhancing the MT structure in VVC by proposing an early termination scheme for the TT split mode. This technique does away with the BT mode that might otherwise cause quality degradation if the blocks extend beyond the picture boundary. Our approach consists of three steps that guide the development of the algorithm. The flowchart in Figure 3 represents the three key steps of the algorithm: (1) the early termination of CU division through the directional gradient from the Scharr operator, (2) the determination of the direction of CU partition by using this gradient, and (3) the extraction of subblock dissimilarities so as to determine the early skip of the TT partitioning process.

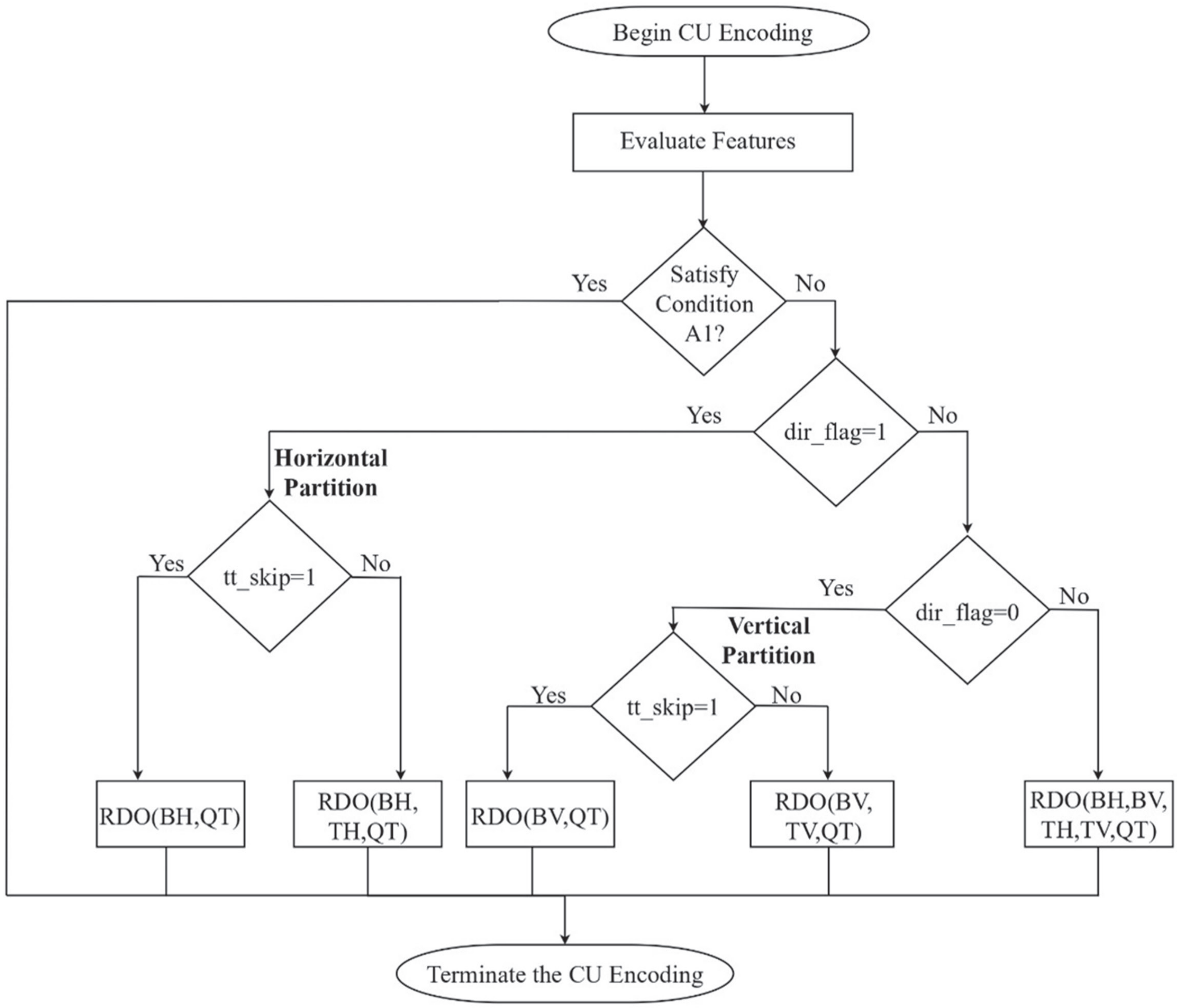



**Fig. 3.** Flow Chart

First, calculate the gradient features by using the Scharr operator in accordance with equations (2), (3), (4), and (5). Then, the directional Scharr-based gradient of the current frame is used as a threshold to assess the texture complexity of CU. There is no necessity to further divide the CUs if their gradient in both horizontal and vertical directions is below this threshold, which means the texture is smooth. On the other hand, if the texture is complex, then it proceeds to the selection of a subsequent partitioning mode. These criteria of assessing the texture complexity are summarized in Equation (16).

$$\text{Condition A1: } G_h < G_{hcf} \quad G_v < G_{vcf} \tag{16}$$

Where $G_{hcf}$ is the average horizontal gradient and $G_{vcf}$ is the average vertical gradient of the current frame.

In Step 2, for the CUs not satisfying Condition A1, we check the partition direction based on the gradient features extracted by the Scharr operator. The horizontal and vertical gradients of the pixels within the current CU determine the partition scheme. The optimal partition direction flag is denoted by the symbol "*dir_flag*", which is formally defined as shown in Equation (17).

$$dir\_flag = \begin{cases} 0, if \ \frac{Gh}{Gv} > Thd \\ 1, if \ \frac{Gh}{Gv} < Thd \\ -1, else \end{cases} \tag{17}$$

The average horizontal gradient and vertical gradient of the current CU are represented by $Gh$ and $Gv$ in the *dir_flag* formulation respectively. By default, the threshold $Thd$ is set to one. When *dir_flag* is equal to 0, indicating a significant horizontal texture change, the CU is divided into two sub-CUs vertically without RDO processing for the horizontal partition mode. On the other hand, when *dir_flag* is equal to 1, indicating a significant vertical texture change, the CU is divided into two sub-CUs horizontally without RDO processing for the vertical partition mode. In the case where *dir_flag* is -1, the CU is RDO processed for all QTMT partition modes until the best mode is selected similar to conventional VTM encoding behavior.

In Step 3, when *dir_flag* is not equal to -1, meaning horizontal or vertical partitioning, the algorithm calculates the dissimilarities in the direction decided by Step 2. This calculation decides whether or not the ternary partitioning can be skipped. The ternary partition skip flag, *tt_skip_flag*, is set based on the decision using Equation (18).

$$tt_{skip} = \begin{cases} 1, if \begin{pmatrix} Dis_{bh} > Dis_{th1}, Dis_{th2} \\ or \ Dis_{bv} > Dis_{tv1}, Dis_{tv2} \end{pmatrix} \\ 0, else \end{cases} \tag{18}$$

The CU determines whether or not to bypass the horizontal or the vertical ternary partition when *tt_skip* is 1. It uses the dissimilarity feature, $Dis$, representing the extent of structural changes between the boundaries of adjacent subblocks and defined by equation (15).

## 4. EXPERIMENTAL OUTCOMES AND ANALYSIS

### 4.1. EXPERIMENTAL OUTCOMES

In this paper, the suggested approach is evaluated on 15 representative video sequences selected from the JVET Common Test Conditions (CTC). The following resolution categories are taken into consideration: Class A (3840×2160), Class B (1920×1080), Class C (832×480), Class D (416×240), and Class E (1280×720). The test sequences were encoded at three different frame rates: 30 Frames per Second (FPS) for Campfire, RaceHorsesC, and RaceHorses; 50 FPS for ParkRunning3, BasketballDrive, Cactus, PartyScene, BasketballDrill, and BlowingBubbles; and 60 FPS for BQTerrace, BQMall, BQSquare, Johnny, KristenAndSara, and FourPeople. The encoding was done in the All-Intra (AI) mode for QP=22, 27, 32, and 37. The BDBR [33] and Average Time Saving ($\Delta T$) are complementary metrics in the scope of VVC, where BDBR quantifies the improvement in encoding efficiency, while $\Delta T$ measures the reduction of computational time needed to perform the coding. Performance has been evaluated based on BDBR and $\Delta T$ using Equation (19).

$$\Delta T = \left(\frac{T_{VTM} - T_{Proposed}}{T_{VTM}}\right) \times 100\% \tag{19}$$

In the context of identical video sequences and identical encoding settings, $T_{VTM}$ refers to the total encoding time of the reference method in VTM-21.2, while $T_{Proposed}$ represents that of the suggested method. Usually, fast algorithms reduce the encoding time significantly; the BDBR values are often higher because of their lower coding efficiency. The average ΔT value of all sequences measures the ability of the fast method to reduce the encoding complexity, while the average BDBR reflects the reduction in coding efficiency due to the faster algorithm. Higher BDBR means a larger reduction in coding quality, and higher time savings mean the encoding complexity has been reduced more significantly.

### 4.2. CODING PERFORMANCE OF THE PROPOSED METHOD

Table 1 presents a comparison of the coding efficiency of the proposed method and VTM-21.2. The original VTM encoder was used for reference, and variations with settings TT_OFF, QT_OFF and BT_OFF were considered. Of note, important configurations include the TT OFF (disabling the TT partition), QT_OFF (without the QT partition), and BT_OFF (without the BT partition). It should be noted that for some CU sizes, such as 128 × 128 and 64 × 64, partitioning is allowed only for QT or NS; without the QT partition, the coding process stops prematurely. Moreover, after MTT partition, for 32 × 32 CUs, the QT partitioning is not allowed anymore. These findings provide insight into the impact of such settings on encoding efficiency and give an overview of possible progress compared with traditional methods.

Table 1 shows the effect of split prohibitions on encoding efficiency. Limiting the TT split by VTM reduces the encoding time by 61.42%, with a 2.57% increase in BDBR. Similarly, QT split blocking results in a time reduction of 24.35% with a 1.97% increase in BDBR, while BT split prohibition offers substantial time savings, as high as 71.09%, with an average BDBR increase of 5.10%. In contrast, the suggested algorithm attains a 46.45% decrease in time with a mere 1.27% increase in BDBR, demonstrating a better optimization between execution time and complexity.

Table 2 distinguishes the efficiency of the suggested method from the most recent fast algorithms.The coding efficiency of the suggested method is assessed by contrasting it with three fast CU partitioning techniques for VVC: [7], [26], and [14].

We focus on contrasting our data with the existing data because the test sequences and their outcomes differ. With a mere rise of 1.23% in BDBR, the analysis indicates that the suggested approach decreases the encoding time by an average of 45.79% when compared with VTM-21.2. Block boundaries were identified in [7] using a block-level Canny edge detector to prevent unnecessary horizontal or vertical splits. In contrast, the proposed method uses Scharr-based gradient features to better capture texture directionality and a dissimilarity (*Dis*) metric to skip unnecessary ternary partitions. The proposed algorithm attains an increment of 10.98% in ΔT, with only a 0.49% increment in BDBR when compared to [7], reflecting improved encoding time.

The approach in [26] used a DenseNet-based predictor for determining the probability of the partition boundary within each 64 × 64 block to reduce the redundant RD computations. Unlike this deep learning-based method, our strategy directly analyzes the structural similarity between adjacent subblocks to determine whether to skip the horizontal or vertical splits. This design brings an improvement in coding quality with a 0.38% reduction in BDBR and enhances the efficiency due to a 1.02% increase in time saving, hence leading to the faster and consistent encoding results.

In [14], SMAD is utilized to decide about the texture complexity and remove the redundant partition modes. It dynamically adjusted the thresholds based on QP for a better trade-off between coding efficiency and complexity. The proposed method made the decision more direct and adaptive by using gradient-based texture analysis and dissimilarity-driven partition control. The highest ΔT of the proposed algorithm is 15.37% higher than that in [13], while the BDBR is increased only by 0.7%, which shows an optimum between computational efficiency and coding performance.

Fig. 4 depicts the RD curve evaluation between the anchor VTM encoder and the suggested approach corresponding to various test sequences. It is evident from the close overlap of RD curves that the suggested technique retains high coding quality, similar to that of the baseline VTM encoder.

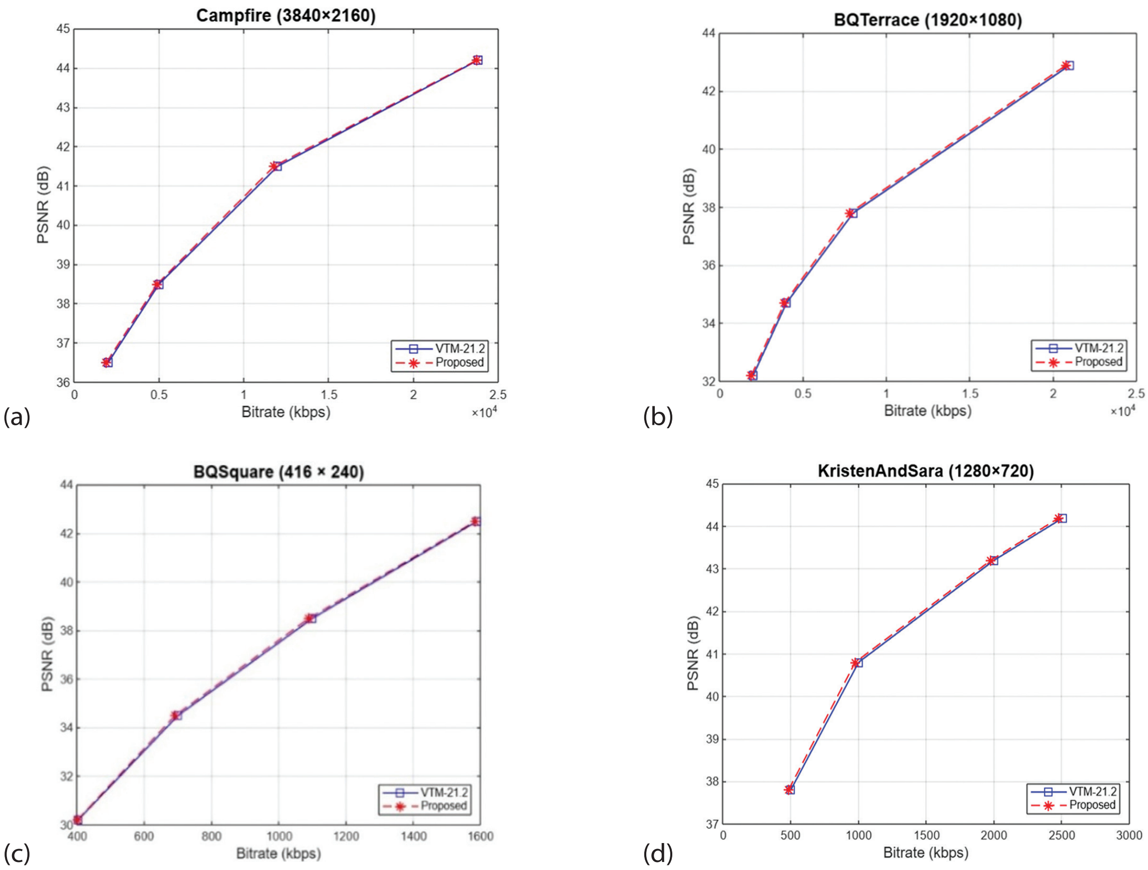



**Fig. 4.** Performance Evaluation of RD Curves on Four Test Sequences: (a) Class A, Campfire (3840×2160), (b) Class B BQTerrace (1920×1080) (c) Class D, BQSquare (416 ×240) (d) Class E, KristenAndSara (1280×720)

## 5. CONCLUSION

This paper introduces an early decision-making approach to CU partitioning for VVC intraframe encoding that depends on only two features, aiming to reduce the high level of computational complexity related to the QTMT framework. This work utilizes the Scharr operator to derive horizontal and vertical gradient information from pixel variation trends, which we then use to guide MT split and early termination. Additionally, we use the dissimilarity feature calculated using SSIM to analyze the edges of adjacent subblocks to characterize the CU structural information and develop an early decision mechanism to skip ternary partitioning. The experimental results indicate that our approach averages a 45.79% reduction in encoding time compared to VTM-21.2, with a minor increase of BDBR by 1.23%. The suggested approach performs well compared to the experimental outcomes of the existing algorithms.

**Table 1.** Performance Metrics Comparison between Proposed Method and VTM-21.2

| Class | Test-Sequence | VTM(TT_OFF) | | VTM(QT_OFF) | | VTM(BT_OFF) | | Proposed | |
|---|---|---|---|---|---|---|---|---|---|
| | | BD-BR (%) | ΔT (%) | BD-BR (%) | ΔT (%) | BD-BR (%) | ΔT (%) | BD-BR (%) | ΔT (%) |
| B | BasketballDrive | 1.91 | 60.71 | 0.61 | 19.25 | 4.03 | 67.45 | 1.18 | 48.89 |
| | BQTerrace | 1.96 | 59.86 | 1.88 | 21.19 | 3.99 | 73.25 | 1.24 | 46.64 |
| | Cactus | 2.49 | 60.55 | 1.11 | 27.58 | 4.16 | 72.87 | 1.26 | 47.68 |
| | Kimono1 | 1.57 | 62.39 | 0.16 | 21.93 | 2.72 | 68.46 | 0.98 | 48.04 |
| | ParkScene | 3.66 | 58.9 | 1.28 | 16.33 | 5.41 | 65.81 | 1.57 | 45.93 |
| C | PartyScene | 1.48 | 61.83 | 2.86 | 23.65 | 4.44 | 75.94 | 0.57 | 42.25 |
| | BQMall | 2.96 | 62.63 | 1.94 | 26.84 | 5.36 | 73.03 | 1.46 | 47.73 |
| | RaceHorsesC | 1.82 | 62.62 | 1.07 | 31.84 | 3.39 | 73.02 | 0.97 | 46.84 |
| | BasketballDrill | 4.09 | 62.63 | 3.81 | 19.69 | 7.98 | 72.39 | 2.15 | 46.84 |
| D | BQSquare | 1.51 | 59.74 | 5.55 | 20.51 | 6.61 | 72.66 | 0.52 | 42.78 |
| | BlowingBubbles | 2.05 | 62.59 | 2.36 | 29.05 | 4.94 | 74.86 | 0.64 | 44.97 |
| | RaceHorses | 2.31 | 60.91 | 2.75 | 27.13 | 5.28 | 74.22 | 0.79 | 44.46 |
| | BasketballPass | 2.92 | 61.05 | 2.11 | 29.61 | 6.38 | 70.67 | 1.25 | 42.88 |
| E | Johnny | 3.39 | 60.84 | 1.19 | 23.02 | 5.76 | 65.87 | 2.18 | 49.74 |
| | KristenAndSara | 3.13 | 61.53 | 1.85 | 23.62 | 5.64 | 66.97 | 1.65 | 46.82 |
| | FourPeople | 3.88 | 64.01 | 1.13 | 28.4 | 5.52 | 70.04 | 2.04 | 50.73 |
| **Average** | | **2.57** | **61.42** | **1.97** | **24.35** | **5.10** | **71.09** | **1.27** | **46.45** |

**Table 2.** Evaluation of the Proposed Algorithm Against Recent Algorithms

| Class (Resolution) | Test-Sequence | Tang2019 [7] | | Zhang2021 [26] | | Song2022 [14] | | Proposed | |
|---|---|---|---|---|---|---|---|---|---|
| | | BD-BR (%) | Δ T (%) | BD-BR (%) | Δ T (%) | BD-BR (%) | Δ T (%) | BD-BR (%) | Δ T (%) |
| A (UHD) | Campfire | --- | --- | 2.21 | 71.34 | 0.59 | 29.28 | 1.09 | 45.63 |
| | ParkRunning3 | --- | --- | 0.85 | 69.85 | 0.27 | 28.34 | 0.73 | 34.81 |
| B (FHD) | BasketballDrive | 0.93 | 49.89 | 2.07 | 62.73 | 0.74 | 34.48 | 1.18 | 48.89 |
| | BQTerrace | 0.81 | 35.29 | 1.52 | 51.61 | 0.62 | 28.85 | 1.24 | 46.64 |
| | Cactus | 0.78 | 37.31 | 1.93 | 58.98 | 0.61 | 30.73 | 1.26 | 47.68 |
| C (WVGA) | PartyScene | 0.34 | 33.18 | 0.76 | 27.34 | 0.42 | 31.62 | 0.57 | 42.25 |
| | BQMall | 0.60 | 33.21 | 1.58 | 36.82 | 0.65 | 33.79 | 1.46 | 47.73 |
| | RaceHorsesC | 0.65 | 30.53 | 1.02 | 39.39 | 0.46 | 27.83 | 0.97 | 46.84 |
| | BasketballDrill | 1.30 | 29.62 | 2.28 | 34.40 | 0.40 | 26.55 | 2.15 | 46.84 |
| D (WQVGA) | BQSquare | 0.22 | 26.02 | 0.41 | 17.94 | 0.29 | 29.97 | 0.52 | 42.78 |
| | BlowingBubbles | 0.23 | 27.71 | 0.42 | 16.60 | 0.43 | 29.34 | 0.64 | 44.97 |
| | RaceHorses | 0.30 | 26.21 | 0.66 | 23.21 | 0.48 | 27.84 | 0.79 | 44.46 |
| E (720p) | Johnny | 1.29 | 40.72 | 3.32 | 55.54 | 0.69 | 30.65 | 2.18 | 49.74 |
| | KristenAndSara | 0.94 | 40.61 | 2.42 | 50.79 | 0.59 | 31.38 | 1.65 | 46.82 |
| | FourPeople | 1.18 | 42.29 | 2.69 | 54.98 | 0.78 | 35.63 | 2.04 | 50.73 |
| **Average** | | **0.74** | **34.81** | **1.61** | **44.77** | **0.53** | **30.42** | **1.23** | **45.79** |

## REFERENCES


[1] G. J. Sullivan, J. Ohm, W. Han, T. Wiegand, "Overview of the High Efficiency Video Coding (HEVC) Standard", IEEE Transactions on Circuits and Systems for Video Technology, Vol. 22, No. 12, 2012, pp. 1649-1668.

[2] W. Hamidouche, T. Biatek, M. Abdoli, E. François, F. Pescador, M. Radosavljević, M. Raulet, "Versatile Video Coding Standard: A Review from Coding Tools to Consumers Deployment", IEEE Consumer Electronics Magazine, Vol. 11, No. 5, 2022, pp. 10-24.

[3] W. Menasri, A. Skoudarli, "Performance comparison of throughput between AVC, HEVC and VVC hardware CABAC decoder," Journal of Real-Time Image Processing, Vol. 20, No. 2, 2023, p. 26.

[4] N. Ling, C. C. J. Kuo, G. J. Sullivan, D. Xu, S. Liu, H. M. Hang, J. Liu, "The Future of Video Coding", APSIPA Transactions on Signal and Information Processing, Vol. 11, No. 1, 2022, pp.1-29.

[5] Y.-W. Huang, "Block Partitioning Structure in the VVC Standard", IEEE Transactions on Circuits and Systems for Video Technology, Vol. 31, No. 10, 2021, pp. 3818–3833.

[6] M. Saldanha, G. Sanchez, C. Marcon, L. Agostini, "Complexity Analysis of VVC Intra Coding", Proceedings of the IEEE International Conference on Image Processing, Abu Dhabi, UAE, 25-28 October 2020, pp. 3119-3123.

[7] N. Tang, J. Cao, F. Liang, J. Wang, H. Liu, X. Wang, X. Du, "Fast CTU Partition Decision Algorithm for VVC Intra and Inter Coding", Proceedings of the IEEE Asia Pacific Conference on Circuits and Systems, Bangkok, Thailand, 11-14 November 2019, pp. 361-364.

[8] J. Chen, H. Sun, J. Katto, X. Zeng, Y. Fan, "Fast QTMT Partition Decision Algorithm in VVC Intra Coding Based on Variance and Gradient", Proceedings of the 2019 IEEE Visual Communications and Image Processing, Sydney, Australia, 1-4 December 2019, pp. 1-4.

[9] J. Cui, T. Zhang, C. Gu, X. Zhang, S. Ma, "Gradient-Based Early Termination of CU Partition in VVC Intra Coding", Proceedings of the IEEE Data Compression Conference, Snowbird, UT, USA, 24-27 March 2020, pp. 103-112.

[10] Y. Chen, L. Yu, H. Wang, T. Li, S. Wang, "A novel fast intra mode decision for versatile video coding," Journal of Visual Communication and Image Representation, Vol. 71, 2020, p. 102849.

[11] Y. Fan, H. Sun, J. Katto, J. Ming, "A Fast QTMT Partition Decision Strategy for VVC Intra Prediction", IEEE Access, Vol. 8, 2020, pp. 107900-107911.

[12] Y. Li, G. Yang, Y. Song, H. Zhang, X. Ding, D. Zhang, "Early Intra CU Size Decision for Versatile Video Coding Based on a Tunable Decision Model", IEEE Transactions on Broadcasting, Vol. 67, 2021, pp. 710-720.

[13] Q. Zhang, T. Cui, L. Huang, B. Jiang, J. Zhao, "Low-complexity intra coding scheme based on Bayesian and L-BFGS for VVC," Digital Signal Processing, Vol. 127, 2022, Art. no. 103539.

[14] Y. Song, B. Zeng, M. Wang, Z. Deng, "An efficient low-complexity block partition scheme for VVC intra coding," Journal of Real-Time Image Processing, Vol. 19, No. 1, 2022, pp. 161-172.

[15] R. Yang, X. He, S. Xiong, Z. Zhao, H. Chen, "Fast CU partition strategy based on texture and neighboring partition information for versatile video coding intra coding," Multimedia Tools and Applications, Vol. 83, No. 10, 2024, pp. 28323-28340.

[16] S. Zhao, X. Shang, G. Wang, H. Zhao, "A fast algorithm for intra-frame versatile video coding based on edge features", Sensors, Vol. 23, No. 13, 2023, p. 6244.

[17] F. Chen, Y. Ren, Z. Peng, G. Jiang, X. Cui, "A fast CU size decision algorithm for VVC intra prediction based on support vector machine," Multimedia Tools and Applications, Vol. 79, No. 37, 2020, pp. 27923-27939.

[18] G. Wu, Y. Huang, C. Zhu, L. Song, W. Zhang, "SVM Based Fast CU Partitioning Algorithm for VVC Intra Coding", Proceedings of the IEEE International Symposium on Circuits and Systems, Daegu, South Korea, 23-26 May 2021, pp. 1-5.

[19] Q. He, W. Wu, L. Luo, C. Zhu, H. Guo, "Random Forest Based Fast CU Partition for VVC Intra Coding", Proceedings of the IEEE International Symposium

on Broadband Multimedia Systems and Broadcasting, Chengdu, China, 18-21 August 2021, pp. 1-4.

[20] T. Li, M. Xu, R. Tang, Y. Chen, Q. Xing, "DeepQTMT: A Deep Learning Approach for Fast QTMT-Based CU Partition of Intra-Mode VVC", IEEE Transactions on Image Processing, Vol. 30, 2021, pp. 5377-5390.

[21] G. Ding, X. Lin, J. Wang, D. Ding, "Accelerating QT-MT-based CU partition and intra mode decision for versatile video coding", Journal of Visual Communication and Image Representation, Vol. 94, 2023, p. 103832.

[22] Z. Jin, P. An, L. Shen, "Fast QTBT Partition Algorithm for JVET Intra Coding Based on CNN", Proceedings of the Pacific Rim Conference on Multimedia, Hefei, China, 27-30 September 2017, pp. 59-69.

[23] G. Tang, M. Jing, X. Zeng, Y. Fan, "Adaptive CU Split Decision with Pooling-Variable CNN for VVC Intra Encoding", Proceedings of the IEEE Visual Communications and Image Processing, Sydney, Australia, 1-4 December 2019, pp. 1-4.

[24] J. Zhao, Y. Wang, Q. Zhang, "Adaptive CU split decision based on deep learning and multifeature fusion for H.266/VVC," Scientific Programming, Vol. 2020, No. 1, 2020, p. 8883214.

[25] M. Amna, W. Imen, S. F. Ezahra, A. Mohamed, "Fast intra-coding unit partition decision in H.266/FVC based on deep learning," Journal of Real-Time Image Processing, Vol. 17, No. 6, 2020, pp. 1971-1981.

[26] Q. Zhang, R. Guo, B. Jiang, R. Su, "Fast CU Decision-Making Algorithm Based on DenseNet Network for VVC", IEEE Access, Vol. 9, 2021, pp. 119289-119297.

[27] B. Abdallah, F. Belghith, M. A. Ben Ayed, N. Masmoudi, "Fast QTMT decision tree for versatile video coding based on deep neural network," Multimedia Tools and Applications, Vol. 81, No. 29, 2022, pp. 42731-42747.

[28] M. Saldanha, G. Sanchez, C. Marcon, L. Agostini, "Configurable fast block partitioning for VVC intra coding using light gradient boosting machine," IEEE Transactions on Circuits and Systems for Video Technology, Vol. 32, No. 6, 2021, pp. 3947-3960.

[29] T. Zhao, Y. Huang, W. Feng, Y. Xu, S. Kwong, "Efficient VVC Intra Prediction Based on Deep Feature Fusion and Probability Estimation", IEEE Transactions on Multimedia, Vol. 25, 2022, pp. 6411-6421.

[30] Y. Huang, J. Yu, D. Wang, X. Lu, F. Dufaux, H. Guo, C. Zhu, "Learning-based fast splitting and directional mode decision for VVC intra prediction", IEEE Transactions on Broadcasting, Vol. 70, No. 2, 2024, pp. 681-692.

[31] F. Liu, J. Wang, Q. Zhang, "ResNet-Driven Joint Decision-Making for VVC Optimization via Gradient Search," in IEEE Access, Vol. 13, 2025, pp. 107442-107454.

[32] Z. Wang, A. C. Bovik, H. R. Sheikh, E. P. Simoncelli, "Image Quality Assessment: From Error Visibility to Structural Similarity," IEEE Transactions on Image Processing, Vol. 13, No. 4, 2004, pp. 600-612.

[33] G. Bjontegaard, "Calculation of average PSNR differences between RD-curves," ITU-T SG16, VCEG-M33, Technical Report, 2001.