Configurable Fast Block Partitioning for VVC Intra Coding Using Light Gradient Boosting Machine

Mário Saldanha, Gustavo Sanchez, *Member, IEEE*, César Marcon, *Senior Member, IEEE*, and Luciano Agostini, *Senior Member, IEEE*

***Abstract*— This article presents a configurable fast block partitioning decision for Versatile Video Coding (VVC) intra-frame prediction using Light Gradient Boosting Machine (LGBM). VVC further improves the coding efficiency by introducing a Quadtree with nested Multi-Type Tree (QTMT), enabling five split types allowing square and rectangular Coding Unit (CU) sizes. However, this improvement in the coding efficiency comes at the cost of a high computational burden since several combinations of block sizes and prediction modes are evaluated through the costly Rate-Distortion Optimization (RDO) process. In this article, we propose a partitioning decision using LGBM classifiers to avoid the exhaustive RDO process and skip the evaluation of split types that are unlikely to be chosen as the best one. For this purpose, five classifiers (one for each split type) were offline trained with an efficient training process and using effective features of texture, coding, and context information. The proposed solution is highly configurable and can provide several operation points with different tradeoffs between timesaving and coding efficiency, according to the application requirements. Considering five operation points, the configurable solution can reduce the encoding time from 35.22% to 61.34%, with coding efficiency losses from 0.46% to 2.43%. Compared to the state-of-the-art, our solution is able to outperform the related works in terms of combined rate-distortion and timesaving.**

***Index Terms*— VVC, intra coding, timesaving, machine learning, light gradient boosting machine.**

I. Introduction

THE massive traffic of digital video over the internet, along with the increasingly widespread of emerging video applications such as Ultra-High Definition (UHD), High-Dynamic Range (HDR), Augmented Reality (AR), and Virtual Reality (VR), has occasioned enormous pressure on the available bandwidth of the telecommunication infrastructures. This became even more noticeable in the COVID-19 pandemic scenario, where the multimedia content consumption over the internet increased a lot, forcing streaming providers to reduce the video quality to support the current demand [1]. Additionally, in a world dominated by battery-powered embedded devices, which have limited processing and energy resources, the high complexity of digital video encoding and decoding is a crucial issue to be addressed to enable real-time processing with low energy consumption.

Remarkable advances in video compression were achieved with the standardization of H.264/MPEG-4 Advanced Video Coding (AVC) [2] and High-Efficiency Video Coding (HEVC) [3], making it possible to encode high-resolution videos efficiently. However, AVC and HEVC do not provide satisfactory performance to reach the coding efficiency required by the current video applications and industry requirements. Thus, creating the demand for new video coding technologies with additional capabilities and pushing the international organizations to establish new video coding standards.

Versatile Video Coding (VVC) [4] is the most recent video coding standardized by the ITU-T and ISO/IEC organizations. This standard was developed by the Joint Video Experts Team (JVET), which was founded in a collaboration between ITU-T Video Coding Experts Group (VCEG) and ISO/IEC Moving Picture Experts Group (MPEG). The primary objective of VVC is to provide a compression efficiency significantly higher than HEVC. At the same time, VVC includes design features that make it highly versatile for various types of video content and applications, such as HDR, screen content, 360° video, and resolution adaptivity.

VVC follows the general approach of block-based hybrid video coding that splits each frame into smaller blocks and processes each block applying intra- or inter-frame prediction. The resulting residual blocks are processed by transform, quantization, and entropy coding. Although the VVC coding structure is similar to its predecessors, it includes several novel techniques and enhancements that result in substantially higher compression efficiency. These improvements include larger block sizes, more flexible block partitioning through Quadtree with nested Multi-type Tree (QTMT) [5], Affine Motion Compensation (AMC) [6], Multiple Transform Selection (MTS) [7], Low-Frequency Non-Separable Transform (LFNST) [8], new intra-frame prediction tools, among others.

One of the most outstanding improvements in comparison to HEVC is the new block partitioning structure. In addition to Quadtree (QT) adopted by HEVC, VVC allows more flexible partition shapes with Binary Tree (BT) and Ternary Tree (TT), enabling rectangular block sizes for mode selection, intra-

Manuscript received August 4, 2021; accepted August 26, 2021. Date of publication August 30, 2021; date of current version June 6, 2022. This work was supported in part by Fundação de Amparo à Pesquisa do Estado do Rio Grande do Sul (FAPERGS), in part by Conselho Nacional de Desenvolvimento Científico e Tecnológico (CNPq), and in part by Coordenação de Aperfeiçoamento de Pessoal de Nível Superior (CAPES) (Finance Code 001), Brazilian research support agencies. This article was recommended by Associate Editor Y. Yang. *(Corresponding author: Mário Saldanha.)*

Mário Saldanha and Luciano Agostini are with the Graduate Program in Computer Science (PPGC) and Video Technology Research Group (ViTech), Federal University of Pelotas, Pelotas 96010-610, Brazil (e-mail: mrdfsaldanha@inf.ufpel.edu.br; agostini@inf.ufpel.edu.br).

Gustavo Sanchez is with the Department of Informatics, IF Farroupilha, Alegrete 97555-000, Brazil (e-mail: gustavo.sanchez@iffarroupilha.edu.br).

César Marcon is with the Graduate Program in Computer Science (PPGCC), Pontifical Catholic University of Rio Grande do Sul, Porto Alegre 90619-900, Brazil (e-mail: cesar.marcon@pucrs.br).

Color versions of one or more figures in this article are available at https://doi.org/10.1109/TCSVT.2021.3108671.

Digital Object Identifier 10.1109/TCSVT.2021.3108671

and inter-frame prediction, and transform coding. However, the coding efficiency improvement significantly impacts the encoding complexity since more split types are evaluated in the costly Rate-Distortion Optimization (RDO) process.

Another important improvement is related to the intra-frame prediction. VVC included many new tools and improved the HEVC tools intending to increase the encoding efficiency. VVC introduces the dual-tree coding structure and supports 17 and 19 Coding Unit (CU) sizes (including square and rectangular shapes) for luminance and chrominance components, respectively. HEVC uses the single-tree structure and only supports five squared CU sizes for both components. VVC supports 65 angular modes instead of the 33 supported by HEVC [9]. New tools, like Multiple Reference Line (MRL) prediction [10], Wide-Angle Intra Prediction (WAIP) [11], Matrix-based Intra Prediction (MIP) [12], and Intra Subpartition (ISP) [13], allow VVC to reach higher efficiency in intra-frame prediction, but with impressive additional complexity. Considering the All-Intra (AI) configuration, where only intra prediction is available, Bossen *et al.* [14] reported that the VVC Test Model (VTM) [15] reached an encoding efficiency 25% higher than the HEVC Test Model (HM) while increasing the encoding complexity by more than 26 times.

The impressive increase in the required computational effort boosted the development of novel solutions intending to reduce the coding complexity while maintaining the coding efficiency. In this scenario, machine learning is a promising approach to reduce the computational effort of video coding applications, minimizing the coding efficiency loss. Among the machine learning techniques, the Light Gradient Boosting Machine (LGBM) [16] is a powerful technique that has shown considerable success in a wide range of applications [17]. Besides, LGBM is based on Decision Tree (DT) classifiers that provide high accuracy and hardware-friendly characteristics, which is also relevant in a scenario dominated by video applications running on battery-powered devices. LGBM is highly customizable to the application needs, outperforming other traditional machine learning techniques, such as Support Vector Machine (SVM), single DT, and Random Forest (RF), since LGBM combines gradient descent and boosting techniques, supporting a large set of adjustable hyperparameters [16].

During intra coding, the VTM execution time can be reduced by 92% or 97% when removing the evaluation of rectangular block sizes [18] or directly inferring the partitioning [19], respectively. These results demonstrate that most of the intra coding complexity is related to the block partitioning process, becoming a bottleneck for VVC real-time application development. This fact boosted works focusing on minimizing this complexity using different approaches. These works include statistical-based heuristics [20]–[23] and machine learning-based techniques [24]–[31] to predict the best block size and avoid unnecessary block partition evaluations. Although VVC allows different coding tree structures for luminance and chrominance components, all these works focused on luminance since it represents more than 85% of the encoding complexity [18]. Even though these solutions obtained impressive results, there is still room to improve the tradeoff between coding efficiency and encoding time savings. Besides, since most of those solutions do not properly explore the correlations and features from the encoding process, they do not present stable results for different video contents and resolutions.

This article presents a solution aiming to maximize the VVC intra-frame prediction encoding time savings while minimizing the impact on the encoding efficiency, with three main goals: (i) generate a configurable solution with multi-operation points allowing an adaptive complexity reduction scheme; (ii) generate a stable solution for different video characteristics and resolutions; and (iii) reach better results than the related works regarding the tradeoff between complexity reduction and coding efficiency. The solution presented in this article is focused on the luminance block partitioning decision based on LGBM classifiers. Our solution considers the QTMT block partitioning as a multiple binary classification problem, where an LGBM classifier is trained offline for each split type, and each classifier is responsible for deciding to perform the split type or not, avoiding the evaluation of split types that are unlikely to be chosen as the best one.

The main contributions of this work are:

- A novel and efficient VVC intra prediction complexity reduction solution;
- A configurable multi-operation points complexity reduction scheme allowing the adaptation for different application requirements;
- The use of LGBM to reduce the QTMT block partitioning complexity sustaining the coding efficiency;
- The design of a robust framework to develop an efficient complexity reduction solution;
- The use of novel features to build the machine learning classifiers;
- The use of specialized set of features according to the split type.

The community of circuits and systems for video technology is highly active in the challenge of reduce VVC complexity maintaining coding efficiency. Firstly, some important articles were published presenting the VVC standard, such as [32]–[36]. Then, new solutions were proposed intending to mitigate the VVC complexity. Some works focused on efficient hardware designs, such as [37]–[39], while others focused on algorithm optimizations, such as [25], [40], and [41]. This article presents novel contributions to our community, as presented above. This work explores a machine learning solution to overcome the high computational effort required by VVC. The presented results surpass the related works when considering rate-distortion and timesaving.

All experiments developed in this work use the VTM version 10.0 under AI encoder configuration specified by the JVET experts. We performed these experiments into a server with the Ubuntu 20.04 operating system, AMD Opteron™ Processor 6376, and 128 GB DDR3 memory.

The remainder of this paper is organized as follows. Section II details the VVC block partitioning structure. Section III presents a brief overview of the LGBM classifier. Section IV discusses the related works focusing on reducing the VVC encoder complexity. Section V describes the methodology to develop the proposed solution and details

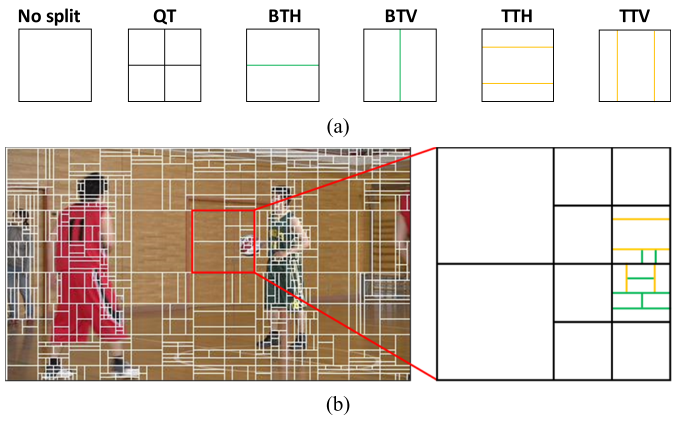


Fig. 1. QTMT partitioning structure. (a) VVC split types and (b) illustration of split results for an encoded video sequence.

the configurable fast CU decision for VVC intra coding. Experimental results and comparisons with the state-of-the-art solutions are presented in Section VI. Finally, Section VII concludes this paper.

II. VVC Block Partitioning Structure

VVC is based on the same block-based hybrid video coding scheme employed in previous standards, such as AVC and HEVC. This scheme splits each video sequence frame into blocks and processes all blocks sequentially by intra- or inter-frame prediction, transform, quantization, and entropy coding.

The VVC standard splits each input frame into Coding Tree Units (CTUs) covering square regions of at most 128 × 128 samples. These larger block sizes supported by VVC can provide a higher compression rate, mainly for high video resolutions. Moreover, each CTU is further recursively partitioned into smaller blocks called Coding Units (CUs).

VVC adopts a coding-tree-based splitting process that, in addition to the HEVC QT splitting, introduces the Multi-Type Tree (MTT) partitioning structure, enabling rectangular CU shapes through binary and ternary splits. This new partitioning structure using quaternary splits followed by binary and ternary divisions is called QTMT.

A CTU is first recursively partitioned with a QT structure. Subsequently, each QT leaf can be further recursively partitioned with an MTT structure using binary and ternary splits. However, when an MTT split is performed, QT split is no longer allowed. The CU sizes may vary from 4 × 4 samples up to 128 × 128 (maximum CTU size), including square and rectangular shapes. The CTU partitioning for luminance and chrominance can be performed jointly, referred to as single-tree, or independently, referred to as dual-tree. The single-tree is employed for P- and B-slices, where both intra- and inter-frame predictions can be applied, whereas the dual-tree is used for I-slices, where only the intra-frame prediction can be performed.

Fig. 1 illustrates the QTMT partitioning structure. Fig. 1(a) shows the six split types available in the QTMT structure. When a CU is defined as no split, the current CU is no further divided, and the coding process is performed with the current CU size; otherwise, a CU can be split with QT, BT, and TT structures. QT splits a CU into four equal-sized square sub-CUs. BT divides horizontally (BTH) or vertically (BTV) a CU into two symmetric sub-CUs. TTH splits a CU into three sub-CUs with the ratio of 1:2:1, and the division also can be performed in horizontal (TTH) and vertical (TTV) directions.

A CU needs to be recursively traversed with all splitting possibilities using the RDO process to select the optimal CU partitioning structure, including no split, QT, BTH, BTV, TTH, and TTV. The best partitioning structure is the one with the smallest RD cost, calculated with (1), where D is the distortion calculated between the original and predicted block, B is the cost in bits of a particular mode, and λ is the Lagrange multiplier.

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

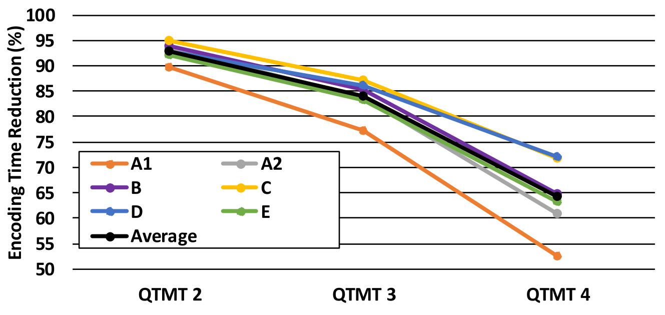
Fig. 1(b) displays the CU size distribution (luminance) for the first frame of the BasketballPass video sequence encoded with Quantization Parameter (QP) 37 using AI configuration. The black, green, and orange lines of the highlighted region denote the QT, BT, and TT divisions, respectively. Since the maximum transform block size supported in VVC is 64 × 64, the intra-frame prediction is carried out only for 64 × 64 or smaller blocks. Thus, VVC forces the first split of all 128 × 128 CUs to be QT in AI configuration.

Fig. 1(b) depicts that the QTMT partitioning provides a very flexible block structure, representing several block sizes and shapes. The smooth regions are encoded with larger blocks and square shapes, whereas more detailed regions are encoded with smaller blocks and rectangular shapes. These block partition types can be adapted to a wide variety of video characteristics, raising the coding efficiency but also increasing the coding complexity.

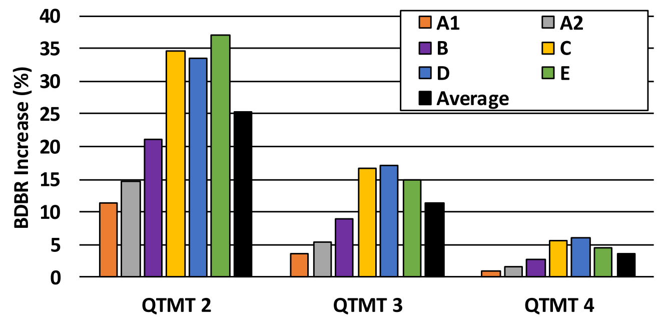
To evaluate the coding complexity and efficiency, Fig. 2 presents the impact of limiting the QTMT depth levels to 2 (QTMT 2), 3 (QTMT 3), and 4 (QTMT 4), considering all classes of video sequences in the Common Test Conditions (CTC) [42] and using AI configuration.

Fig. 2(a) depicts for all cases that the VTM execution time is drastically reduced when the maximum QTMT depth is limited. On average, the encoding time reduces about 93%, 84%, and 64% for maximum QTMT depths 2, 3, and 4, respectively. Nevertheless, Fig. 2(b) shows that limiting the QTMT depth also expressively increases the Bjontegaard Delta Bitrate (BDBR) [43]. On average, BDBR increases by 25%, 11%, and 4% for maximum QTMT depths 2, 3, and 4, respectively.

The impact on the coding efficiency has a clear correlation with the encoded video sequence. The video sequences with high resolution (classes A1, A2, and B) present the lowest BDBR increase since, in most cases, they are encoded with larger block sizes, which are achieved in the first QTMT levels. In contrast, classes C, D, and E, which have videos with low resolutions, show a high BDBR increase since, in most cases, they are better encoded with smaller blocks, which are obtained in the last QTMT levels. Besides, the block size



(a)



(b)

Fig. 2. Impact of limiting the maximum QTMT depth for each class of video resolution: (a) Encoding time reduction and (b) BDBR increase.

has a high correlation with video content and encoder context attributes, such as QP value.

Therefore, reducing the number of splits evaluated in the QTMT structure, correlating the video content, encoder configurations, and other encoder attributes can avoid the high cost of assessing the entire RDO process minimizing the impact on the coding efficiency. We propose a QTMT partitioning decision approach based on a machine learning technique for expressive encoding time savings while maintaining coding efficiency.

III. LGBM Classifiers

Ensemble models in machine learning combine the decisions of multiple weak learners to improve the overall performance of a system [17], providing higher accuracy results than individual models. The two main types of ensemble approaches are *bagging* that creates individual classifiers for taking decisions based on the majority votes of all classifiers, and *boosting*, which builds the classifiers iteratively, minimizing the error of the earlier trained classifiers [17].

LGBM is a gradient boosting framework developed by Microsoft researchers using tree-based learning algorithms [16]. Fig. 3 exemplifies the LGBM training approach that builds a DT ensemble sequentially to minimize losses and improve the model at each iteration step. Each iteration determines a new DT model training concerning the error of the entire ensemble learned so far. The learning rate controls the gradient descent approach used to minimize the loss when adding trees.

LGBM achieves a solid predictive model by combining N tree models (f_1, f_2, $f_3, \ldots, f_n$) and the final result described

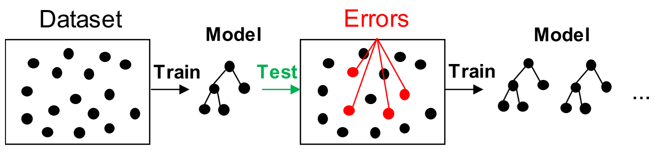


Fig. 3. LGBM training approach.

in (2) aggregates the results from each step.

$$f(x) = \sum_{n=1}^{N} f_n(x) \tag{2}$$

Unlike other tree-based learning algorithms, LGBM grows tree leaf-wise (vertically) since it can reduce the prediction loss more efficiently than algorithms that produce level-wise trees (horizontally). Moreover, conventional implementations of GBMs scan all the data instances to estimate the information gain of all possible split points, which is very time-consuming for the training process. LGBM uses Gradient-based One-Side Sampling (GOSS) and Exclusive Feature Bundling (EFB) to overcome this problem. GOSS and EFB are sampling methods for data selection, discarding some well-trained instances (small training error), and reducing the dimensionality of the features while maintaining high accuracy [16].

In summary, LGBM has many advantages compared to other machine-learning models, such as (i) ability to handle large-scale data, (ii) support of parallel and Graphics Processing Unit (GPU) learning, (iii) low memory usage, (iv) fast training speed, (v) simple implementation with tree-based algorithm (vi) high accuracy, and (vii) low inference time. The last three characteristics are crucial for this work since it aims at reducing the VVC encoding complexity without harming the coding efficiency.

The LGBM techniques provide a highly flexible training process to control the learning rate hyperparameters, dataset sampling, and DT characteristics, generating a high-efficient model when adequately optimized.

IV. Related Works

Different approaches are employed to reduce the complexity of block size decisions in VVC encoders. This section presents relevant works that employ statistical analysis to build fast block size decision heuristics, followed by works that use machine learning approaches and our motivations to employ LGBM classifiers in the block partitioning decision.

Lei *et al.* [20] proposed a fast solution to decide in advance the direction of BT and TT partitions. Their solution evaluates a subset of directional intra-frame prediction modes for virtual subpartitions of the current block to estimate the horizontal and vertical splitting costs of the current block. Based on the estimated costs, their solution can decide by skipping horizontal or vertical partitions. Their approach, implemented in VTM 3.0 and tested with AI configuration, saves 45.8% of the encoding time with a 1.03% BDBR increase.

Cui *et al.* [21] proposed a complexity reduction scheme based on the direction of the sample gradients to decide

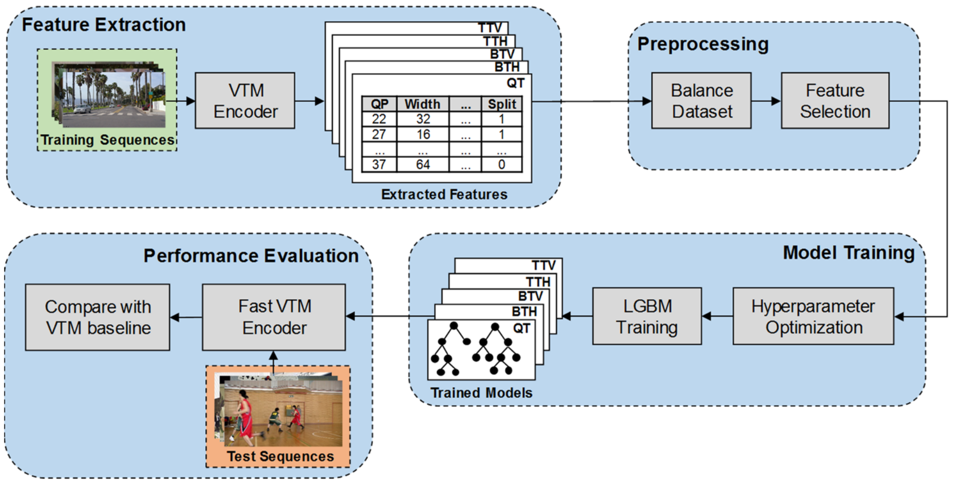


Fig. 4. Framework for training CU partitioning decision with LGBM models and evaluating the performance in the VTM encoder.

the best block partitioning structure. Their scheme performs the decision on three partitioning possibilities, including split or not, horizontal or vertical, and BT or TT. To perform the decision, the gradients of current block subpartitions are computed in four directions and compared with predefined threshold values. Their scheme was implemented in VTM 5.0 and evaluated with AI configuration, reaching 50.01% of encoding timesaving with an increase of 1.23% in the BDBR.

Fan *et al.* [22] presented a solution based on the current block variance, subpartition variances, and Sobel Filter. The current block variance is computed to check the homogeneity of 32×32 blocks and early terminate the QTMT evaluation. The variance of subpartitions is calculated to choose only one split among QT, BT, and TT. The Sobel filter is used to decide by skipping the BT/TT partitions and evaluated only the QT partitioning. Their solution was implemented in VTM 7.0 and evaluated with AI configuration. The experimental results reached a 49.27% encoding timesaving with a 1.63% BDBR increase.

Li *et al.* [23] proposed a complexity reduction solution to skip binary and ternary splitting based on residual block variances of subpartitions, obtained through the absolute difference between original and predicted samples. The absolute difference between variances of vertical and horizontal subpartitions is computed and compared with predefined threshold values to early skip BT and TT evaluations. Their solution was implemented in VTM 7.1 and tested with AI configuration, saving 43.9% of the encoding time with a 1.50% BDBR increase.

Fu *et al.* [24] developed a fast block partitioning algorithm using a classifier based on the Bayesian decision rule. The information derived from the current block and horizontal binary splitting is used as input features for the classifier, responsible for deciding when skipping the vertical split types. Additionally, the horizontal ternary split is skipped if the cost of the vertical binary split is lower than the cost of the horizontal binary split. Their solution was implemented in VTM 1.0 and simulated with AI configuration, obtaining 45% of encoding time reduction and 1.02% of BDBR increase.

Yang *et al.* [25] proposed a complexity reduction scheme composed of a fast block partitioning solution based on DT classifiers and a fast intra mode decision to reduce the number of angular intra-frame prediction modes evaluated. They trained one DT classifier for each split type using the texture information of the current and neighboring blocks. Since the classifiers are responsible for deciding the best split type before performing the prediction with the current block size (i.e., not split type), they used only texture information features of the current and neighboring blocks. The proposed scheme was implemented in VTM 2.0 and evaluated according to AI configuration. The fast block partitioning solution saves 52.59% of encoding time for a 1.56% of BDBR increase. The fast intra mode decision reduces 25.51% of the encoding time and increases 0.54% of the BDBR.

Chen *et al.* [26] developed a complexity reduction solution using SVM classifiers to decide between horizontal and vertical partitioning. Six classifiers are trained online using only texture information of the current block during the first frame encoding; the remaining frames are encoded, applying the decisions of the trained classifiers. The proposed solution was implemented in VTM 2.1 and tested with AI configuration, providing 50.97% encoding timesaving with a BDBR increase of 1.55%.

Amestoy *et al.* [27] proposed a fast block partitioning based on RF classifiers. Three classifiers were trained to decide on split or not, split with QT or BT/TT, and split horizontally or vertically. This technique was implemented in VTM 5.0 and evaluated with Random Access (RA) configuration, saving 30.1% of encoding time with a 0.61% BDBR increase.

Tissier *et al.* [28], Zhao *et al.* [29], and Li *et al.* [30] proposed complexity reduction solutions based on the Convolutional Neural Network (CNN) to define the best block partitioning. The solution proposed in [28] was implemented and evaluated with VTM 6.1 using AI configuration. The experimental results showed 42.2% encoding timesaving with a 0.75% BD-rate increase. The solutions proposed in [29] and [30] were evaluated in VTM 7.0, also using AI configuration. The first solution reduces the encoding time by about

39.39%, with a BD-rate increase of 0.86%; the second one saves 44.65% of encoding time with a 1.32% BD-rate increase.

Considering the tradeoff between encoding timesaving and coding efficiency, the results achieved by machine learning-based solutions outperform statistical-based approaches. Several of these techniques rely on machine learning-based approaches and can enhance the VVC encoder performance. While studying the methodology of machine learning-based works, we noticed some aspects that could still be improved. For instance, the extraction of relevant and effective features for dealing with the split type decision; several of these solutions cannot provide a good tradeoff between encoding timesaving and encoding efficiency and generalize for different video contents and resolutions because of the lack of relevant features in the training process. Besides the use of more efficient machine learning models, the works [25] and [26] use DT and SVM, which in most cases are outperformed by GBM [17]. Additionally, although CNN models provide excellent accuracy results, they need a large dataset to achieve the desired performance and supply a better generalization, and the CNN hardware implementations are still a big challenge.

Therefore, our solution employs well-trained LGBM models with features highly correlated with the split type decision to reach a higher encoding timesaving with negligible impact on the coding efficiency, providing a competitive tradeoff with state-of-the-art solutions.

V. Configurable Fast Coding Unit Partition Decision Based on Machine Learning

Since RDO evaluates several CU partitions and prediction modes to find the best encoding possibility, it is desirable to skip some evaluations in the QTMT structure to reduce the encoding time without compromising the coding efficiency. This section presents the proposed configurable fast QTMT partitioning decision based on LGBM classifiers to avoid the evaluation of some split types in the RDO process. Section V-A details the methodology employed to develop the solution. Section V-B describes the feature analysis and selection, Section V-C details the training and performance of LGBM classifiers, and Section V-D presents the integration of LGBM classifiers with the QTMT splitting process inside the VTM reference software.

A. Methodology

We used data mining to discover strong correlations between the coding context and its attributes for defining machine learning models that determine when to perform a QTMT split type, saving coding time with negligible reduction in coding efficiency. Our solution divides the block partition decision into five binary classification problems instead of creating an LGBM classifier that directly solves the QTMT structure multiclass problem. This approach allows the design of specialized classifiers for each split type, saving expressive encoding time while minimizing the coding efficiency loss. For this purpose, we trained offline an LGBM classifier for each split type, including QT, BTH, BTV, TTH, and TTV, and each classifier decides to skip or not the corresponding split type.

TABLE I
Video Sequences Used for Training

Training sequence	Resolution	Bit depth	Fps
TrafficFlow [44]	3840×2160	10	30
BuildingHall2 [44]	3840×2160	10	50
Kimono1 [44]	1920×1080	8	24
ParkScene [44]	1920×1080	8	24
Vidyo1 [44]	1280×720	8	60
Netflix_DrivingPOV [45]	1280×720	8	60
pedestrian_area [46]	832×480 (downsampled)	8	25
Flowervase [44]	416×240	8	30

Fig. 4 presents the framework used to train and implement the LGBM classifiers in the VTM encoder. A set of video sequences were selected to extract the features and train the classifiers. The VTM encoder was modified to collect several statistical data with relevant information for the CU split decision and generate the dataset of each split type. The datasets are composed of relevant features from the encoded video sequence, encoder attributes, and the split decision. The preprocessing step was performed to balance the datasets and select the most important features. The selected features are used as input for training the classifiers; this step includes hyperparameter optimization and the training of each classifier. The final step consists of evaluating the coding efficiency and encoding timesaving using a modified VTM encoder, which incorporates the LGBM classifiers for deciding the QTMT partitioning instead of the full RDO. In this step, different video sequences from those used in the training phase are evaluated.

Table I presents the eight video sequences used in the training process with resolutions ranging from 416 $\times$ 240 up to 3840 $\times$ 2160 pixels.

These video sequences encompass a wide range of video characteristics (e.g., 8- to 10-bit depth and 24 to 60 frames per second – fps) for rendering several examples of block partitioning decisions in the training process.

The video sequences were encoded following the encoder configurations specified in JVET CTC [42] for AI configuration, using QP values 22, 27, 32, and 37. We extracted the datasets based on 120 frames to reduce the training process complexity; these datasets were balanced according to the number of instances for each frame, block size, QP value, and output class.

B. Feature Analysis and Selection

We collected a large amount of data from the video sequences and internal encoding variables to find features that could lead to effective decisions of CU split type. All these features were extracted directly during the encoding, where additional functions were implemented in the VTM encoder. These features encompass four information categories: CU samples, local samples, context, and coding information.

CU samples information considers features related to the current CU samples. All these features are computed based on luminance samples inside the whole CU, including width and height of the current CU, area, block ratio, variance (*var*), horizontal (*Gx*) and vertical (*Gy*) gradients based on Sobel operator, *Gx* divided by *Gy* (*ratioGxGy*), and the sum of *Gx* and *Gy* divided by the block area (*normGradient*).

The information of local samples refers to features obtained in smaller regions of the current CU, such as the absolute difference of variances on four sub-quarters (*diffVarQT*), maximum variance on four sub-quarters (*maxVarQT*), the absolute difference between variances of upper and lower regions of the CU (*diffVarHor*), and the absolute difference between left and right regions of the CU (*diffVarVer*).

Context information includes features of left, above, above-left, and above-right neighboring CUs, such as average QT (*neighAvgQT*) and MTT (*neighAvgMTT*) depth levels in neighboring CUs and number of neighboring CUs with QT (*neighHigherQT*) and MTT (*neighHigherMTT*) depth levels higher than the current CU.

Since not split is evaluated first than QT, BT, and TT splits, we can consider several coding attributes obtained with the current CU size for deciding the split types. *Coding information* comprises coding attributes related to the current CU evaluated with not split type, such as QP, RD cost (*currCost*), distortion (*currDistortion*), current QT (*QTD*), BT (*BTD*), MTT (*MTTD*), and QTMT (*QTMTD*) depth levels, best intra prediction mode (*currIntraMode*), MRL index (*mrlIdx*), LFNST index (*lfnstIdx*), ISP mode (*ispMode*), and MTS flag (*mtsFlag*). Besides, since the split types are performed in order, the next split can take advantage of information obtained in the split evaluated previously, then the coding information also considers BTH RD cost (*costBTH*), BTV RD cost (*costBTV*), *costBTH* divided by *costBTV* (*ratioCostBTHBTV*), and TTH RD cost (*costTTH*). It is also important to mention that the corresponding RD-cost is unavailable when a previous split type evaluation is skipped, and the feature has the maximum finite double-precision value.

Table II shows the 19, 28, 29, 28, and 29 features used in the QT, BTH, BTV, TTH, and TTV classifiers, selected employing the Feature Selector tool [47]. We discarded collinear and low-importance features to reduce the dataset dimensionality and the computational cost of the training process.

Note that a new set of features is explored in this solution when compared to the related works, such as *currCost, currDistortion, mtsFlag*, *lfnstIdx*, *ispMode*, *mrlIdx*, *costBTH*, *costBTV*, *costTTH*, *ratioCostBTHBTV*, *diffVarQT*, *maxVarQT*, *neighAvgMTT*, and *neighAvgQT*. Besides, unlike the related works that use the same set of features for different decisions, in this solution, a different set of features is considered according to the split type, allowing more accurate results.

Fig. 5 presents the feature importance of the top 10 features for each classifier. The feature importance was measured using the split metric, which calculates the number of times the feature is used in the model. One can notice that features related to RD cost (*currCost* and *currDistortion*) followed by texture information have great importance for all classifiers. Besides, RD cost of previous split types also provides valuable information for the next split evaluations. Features indicating a horizontal texture direction such as *Gx* and *diffVarHor* are most important for horizontal splits (BTH (Fig. 5(b)) and TTH (Fig. 5(d)). While features indicating a vertical texture direction such as *Gy* and *diffVarVer* are most important for vertical splits (BTV (Fig. 5(c)) and TTV (Fig. 5(e)).

TABLE II
FEATURES USED FOR EACH CLASSIFIER

Feature	QT	BTH	BTV	TTH	TTV
QP	×	×	×	×	×
currCost	×	×	×	×	×
currDistortion	×	×	×	×	×
width	×	×	×	×	×
height		×	×	×	×
area		×	×	×	×
blockRatio		×	×	×	×
QTD			×		
BTD		×	×	×	×
MTTD		×	×	×	×
QTMTD		×	×		
currIntraMode	×	×	×	×	×
mrlIdx		×			
ispMode	×	×	×	×	×
mtsFlag		×	×	×	×
lfnstIdx		×	×		
var	×	×	×	×	×
diffVarQT	×	×	×	×	×
maxVarQT	×	×	×	×	×
diffVarVer	×	×	×	×	×
diffVarHor	×	×	×	×	×
Gx	×	×	×	×	×
Gy	×	×	×	×	×
ratioGxGy	×	×	×	×	×
normGradient	×	×	×	×	×
neighAvgQT	×	×	×	×	×
neighHigherQT	×	×	×	×	×
neighAvgMTT	×	×	×	×	×
neighHigherMTT	×	×	×	×	×
costBTH			×	×	×
costBTV				×	×
ratioCostBTHBTV				×	×
costTTH					×
Number of features	**19**	**28**	**29**	**28**	**29**

Fig. 6 exemplifies the probability density of four selected features for QT, BTH, and BTV classifiers. The attributes *currCost* and *currDistortion* demonstrate a clear correlation with the QT split, where low values of these attributes indicate a high probability of not splitting with QT. Also, low values of *diffVarHor* and *diffVarVer* indicate a high probability of not splitting with BTH and BTV, respectively.

C. Classifiers Training and Performance

In the training process of classifiers, a crucial step to maximize the model performance is the hyperparameter optimization. The LGBM brings several hyperparameters to provide higher accuracy and deal with overfitting and underfitting that need to be properly optimized. For this purpose, the hyperparameters of each classifier were optimized using the efficient Optuna framework [48] and applying the Tree-structured Parzen Estimator (TPE) [49] approach.

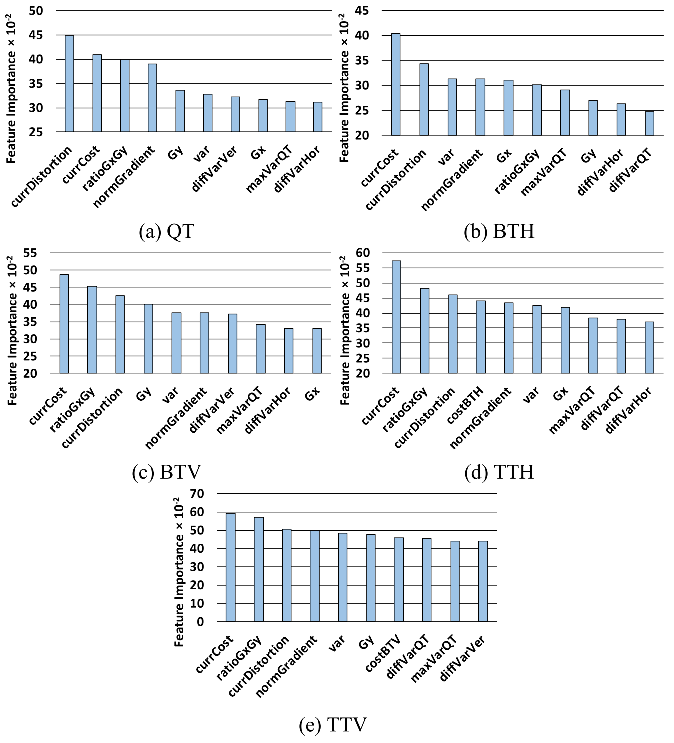


Fig. 5. Feature importance ranking of top 10 features for (a) QT, (b) BTH, (c) BTV, (d) TTH, and (e) TTV classifiers.

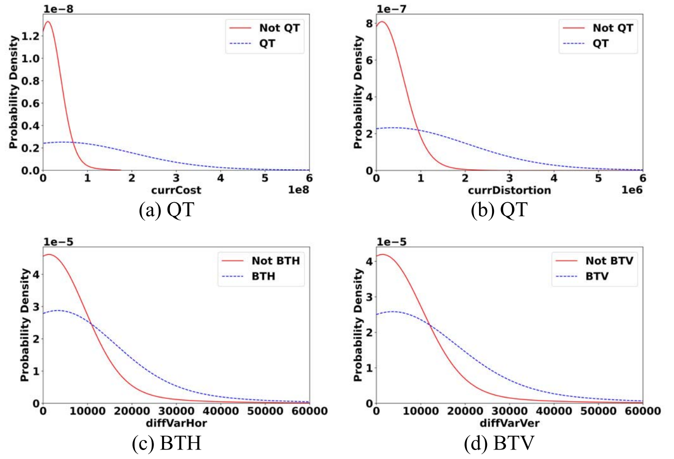


Fig. 6. Probability density functions for (a) and (b) QT, (c) BTH, and (d) BTV classifiers regarding four analyzed attributes.

The main optimized hyperparameters and the best values obtained for each classifier are presented in Table III. *Learning_rate* corresponds to how quickly the error is corrected from each iteration (or tree) to the next. *Feature_fraction* specifies the percentage of features used for each iteration. *Bagging_fraction* refers to the data rate selected when bagging is applied, whereas *bagging_freq* indicates the frequency *k* for performing bagging. *Num_leaves* denotes the maximum number of leaves in one tree, and *Max_depth* limits the maximum depth for each tree. Finally, *Num_iterations* specifies the number of boosting iterations (or the number of trees).

TABLE III
OPTIMIZED HYPERPARAMETERS FOR EACH CLASSIFIER

Hyperparameter	QT	BTH	BTV	TTH	TTV
learning_rate	0.10	0.13	0.12	0.12	0.14
feature_fraction	0.84	0.70	0.81	0.84	0.92
bagging_fraction	0.73	0.71	0.97	0.86	0.98
bagging_freq	3	5	7	7	1
num_leaves	254	250	231	251	253
max_depth	24	60	13	39	34
num_iterations	176	176	256	268	298

TABLE IV
ACCURACY AND F1-SCORE RESULTS FOR EACH CLASSIFIER

Metric	QT	BTH	BTV	TTH	TTV
Accuracy	83.50%	74.69%	74.77%	76.52%	76.52%
F1-score	83.18%	74.64%	74.72%	76.59%	76.58%

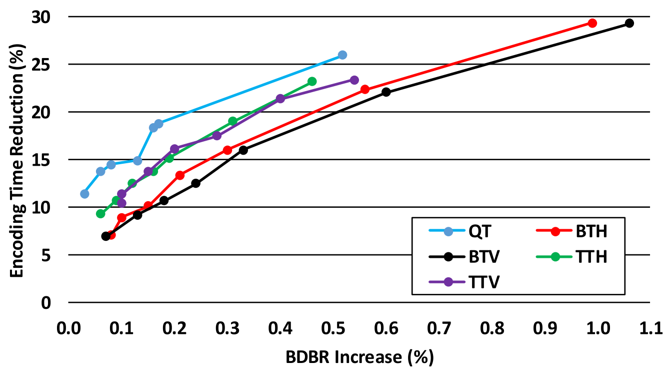


Fig. 7. Encoding time reduction and coding efficiency of each classifier for seven threshold values.

After the hyperparameter optimization process, the classifiers were evaluated using the 10-fold cross-validation, considering accuracy and F1-score metrics. Table IV presents the accuracy and F1-score results for each classifier, demonstrating that the classifiers obtain stable results for both metrics and can provide high performance to predict the CU split type.

The proposed solution follows the hierarchical process of VTM, and it uses the LGBM classifiers to avoid the evaluation of split types that have a low probability of being chosen as optimal partitioning. Since our solution encompasses five classifiers, each classifier indicates a probability value to skip the evaluation of a determined split type. By default, the decision threshold used by the LGBM model is 0.5, and the confidence of prediction is given by how close to 0 or 1 is the decision function output. If the output is higher than 0.5, the classifier decides to skip the split type evaluation; otherwise, the classifier decides to remain the split type evaluation. However, the decision threshold can be configured and different tradeoff results between encoding time reduction and encoding efficiency can be achieved.

Fig. 7 displays the performance of each individual classifier implemented in VTM, regarding encoding time reduction and BDBR impact for the following threshold values: 0.3, 0.4, 0.5,

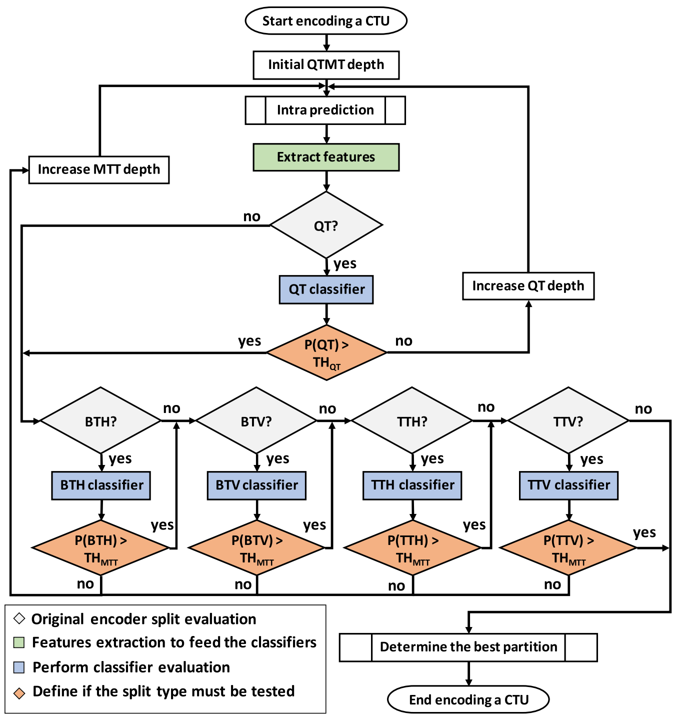


Fig. 8. Flowchart of the proposed solution integrated with the QTMT splitting process.

0.55, 0.6, 0.65, and 0.7. This evaluation allows the analysis of the individual results of each classifier for different operation points to validate their performance in terms of timesaving and BDBR. Note that the lower the threshold value, the higher the encoding time reduction and BDBR impact since more splits are skipped. In contrast, the higher the threshold value, the lower the encoding time reduction and BDBR impact since more splits are evaluated. Therefore, threshold values 0.3 and 0.7 provide the highest and the lowest time savings for all classifiers, respectively.

At this point, it is necessary to highlight that even this first evaluation showed a low impact on the coding efficiency for each classifier regarding different threshold values, the integration of all classifiers is not a trivial task, and some adaptations were needed, as presented in the next section.

D. Classifiers Integration

Fig. 8 presents the flowchart of the proposed solution composed of the five LGBM classifiers integrated with the QTMT splitting process. The white and light gray colors refer to native steps of the VTM encoding flow, and the green, blue, and orange colors represent the new steps of our solution introduced in the encoder, including feature extraction, classifier evaluation, and split evaluation decision, respectively.

After evaluating the intra-frame prediction with the not split type, our solution extracts the features to feed the LGBM classifiers. Subsequently, the VTM encoder verifies the split type that could be evaluated and according to this split type, an LGBM classifier is applied. Each LGBM classifier gives a probability to skip the evaluation of the corresponding split type. The probabilities obtained with the classifiers for QT, BTH, BTV, TTH, and TTV are P(QT), P(BTH), P(BTV), P(TTH), and P(TTV), respectively. These probabilities are compared with decision thresholds to skip the evaluation of split types with a low probability to be chosen as the best one. Then, the evaluation of a determined split type is skipped if the probability is higher than the decision threshold; otherwise, the encoding flow remains without modifications. The VTM split type evaluation is performed sequentially; thus, the proposed solution also performs a sequential decision, as shown in the flowchart. This approach takes advantage of the information of previously evaluated split types with specialized classifiers for each split type, intending to increase the accuracy of the decisions.

The VTM split type evaluation is performed sequentially; thus, the proposed solution also performs a sequential decision, as shown in the flowchart. This approach takes advantage of the information of previously evaluated split types with specialized classifiers for each split type, intending to increase the decisions accuracy.

In the proposed solution, we established two decision thresholds to provide more flexibility: one threshold for QT, called TH_{QT}, and another threshold for MTT (including horizontal and vertical BT/TT partitions), called TH_{MTT}. From our experimental analysis, we have noticed that using these two threshold values provides more flexibility for the proposed solution than using only one threshold value. In contrast, employing different decision thresholds for horizontal and vertical BT/TT partitions did not significantly modify the reached results.

Additionally, we have noticed that when our solution decides to skip all MTT splits in a given direction and the best partitioning would be in that direction, significant coding efficiency loss is caused. Therefore, the proposed solution skips all splits in a given direction (e.g., BTH and TTH) only if all splits have a high probability of being skipped (empirically defined as 0.7); otherwise, the split type with the lowest probability is evaluated. Considering these integration decisions, the QTMT split process defined in Fig. 8 was implemented inside the VTM in substitution of the original VVC intra-frame prediction QTMT split process and the evaluations are presented in the next section.

The use of the decision thresholds makes our solution highly configurable, providing multi-operation points and allowing the adaptation for different application requirements. In our solution, the thresholds are defined before starting the encoding of a video sequence according to the operation point desired. The change of the operation point can be done at multiple levels, according to the user's need, including CTU level, frame level, group of pictures (GOP) level or video level. The experiments presented in the next section consider the last option. Considering the flowchart presented in Fig. 8, the change of operation point is done by changing the TH_{QT} and TH_{MTT} values.

In our solution, different combinations of threshold values can be defined to maximize the encoding timesaving or minimize the coding efficiency loss. On the one hand, increasing the threshold values reduces the number of split types skipped,

TABLE V
VALUES OF TH_{QT} AND TH_{MTT} FOR THE FIVE OPERATION POINTS

Configuration	TH_{QT}	TH_{MTT}
C_1	0.7	0.7
C_2	0.7	0.6
C_3	0.6	0.55
C_4	0.5	0.5
C_5	0.4	0.4

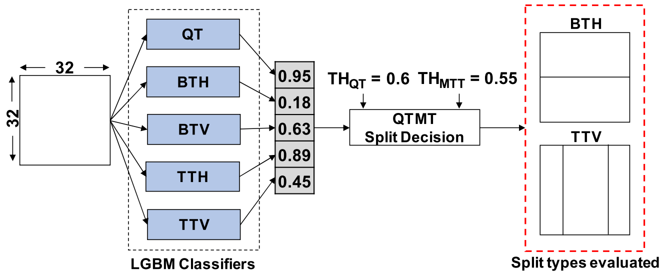


Fig. 9. QTMT split decision using the proposed solution for a 32 × 32 CU.

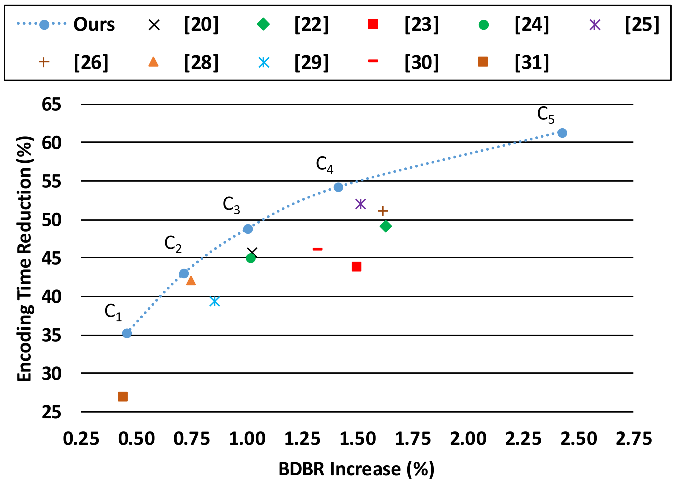


Fig. 10. ETS and BDBR increase for the five operation points of the proposed configurable solution and comparison with the related works.

allowing the encoder to evaluate more split types, and resulting in higher coding efficiency. On the other hand, decreasing the threshold values increases the number of split types skipped, resulting in a higher encoding time reduction.

In the next section, we present the evaluation of the proposed method using a configuration level with five operation points. These operation points were defined through an extensive experimental evaluation, showing good results to support different application requirements. However, the high flexibility of our solution enables even more combinations of threshold values to find the best operation point according to the application requirements. These five operation points are presented in Table V, considering the TH_{QT} and TH_{MTT}.

Fig. 9 exemplifies the proposed solution in the QTMT split decision for a 32 × 32 CU, considering the C_3 operation point. Each classifier provides an output indicating the probability of skipping the associated split type. In the example of Fig. 9, as the classifiers of QT, BTV, and TTH decided by skipping the evaluations, only BTH and TTV split types are evaluated for the current CU since only these splits have a lower probability than the decision threshold values.

VI. EXPERIMENTAL RESULTS AND COMPARISONS WITH RELATED WORKS

This section presents the results of the configurable fast block partitioning decision solution for VVC intra coding using LGBM classifiers. We evaluated all experiments in VTM 10.0 following the CTC specified by JVET for Standard Dynamic Range (SDR) video sequences [42] in AI encoder configuration, which allows only intra-frame prediction. CTC includes six classes of video sequences (A1, A2, B, C, D, and E) with resolutions ranging from 416 × 240 up to 3840 × 2160 pixels.

It is important to highlight that the training process did not use JVET CTC video sequences; consequently, this evaluation considered different video sequences from the ones used in the training step, allowing a robust evaluation of the proposed solution.

Table VI presents the Encoding Time Saving (ETS) and the coding efficiency, measured in BDBR of our solution. These experiments considered the five operation points presented in the previous section and a configuration at the video level, meaning that the operation point does not change during the video sequence encoding. The results are presented for each video sequence, but the average and standard deviation are also presented to demonstrate the robustness of our solution considering different video characteristics and resolutions.

From Table VI, one can observe that, according to the TH_{QT} and TH_{MTT} values, the ETS and BDBR can reach a large range of average values: from an ETS of 35.22% with a BDBR increase of 0.46% to an ETS of 61.34% with a BDBR increase of 2.43%.

The results showed that the proposed method can be efficiently applied to support a wide range of application requirements, with expressive ETS gains and with minor impacts in the BDBR results. These experiments also showed that our solution presented stable results for the evaluated video sequences, presenting low standard deviation for BDBR and ETS results, even considering different video characteristics and resolutions. Besides, the reached results outperform the state-of-the-art solutions in terms of combined rate-distortion and timesaving.

Fig. 10 summarizes the comparisons with the related works. This figure presents a relation between ETS and BDBR for the related works and the five operation points previously defined to our solution. Ten related works were compared with our results, the works of Lei *et al.* [20], Fan *et al.* [22], Li *et al.* [23], Fu *et al.* [24], Yang *et al.* [25], Chen *et al.* [26], Tissier *et al.* [28], Zhao *et al.* [29], Li *et al.* [30], and Park and Kang [31].

TABLE VI
ENCODING TIMESAVING AND CODING EFFICIENCY RESULTS OF THE PROPOSED SOLUTION FOR FIVE OPERATION POINTS

Class	Video sequence	*Configuration Level*									
		C1		*C2*		*C3*		*C4*		*C5*	
		BDBR (%)	ETS (%)	BDBR (%)	ETS (%)	BDBR (%)	ETS (%)	BDBR (%)	ETS (%)	BDBR (%)	ETS (%)
A1	Tango2	0.39	42.53	0.49	48.54	0.71	53.15	1.01	57.19	1.62	61.97
	FoodMarket4	0.35	36.12	0.50	41.28	0.69	46.55	0.96	51.18	1.51	56.82
	Campfire	0.39	29.40	0.59	38.85	0.83	43.76	1.18	51.82	2.02	59.57
A2	CatRobot	0.41	30.39	0.64	38.68	0.90	44.12	1.23	50.81	2.11	57.28
	DaylightRoad2	0.50	38.87	0.84	47.75	1.10	53.67	1.48	59.48	2.45	66.82
	ParkRunning3	0.21	28.70	0.31	35.58	0.45	41.36	0.61	47.60	0.98	52.12
B	MarketPlace	0.23	35.65	0.40	46.98	0.60	54.47	0.84	60.62	1.41	67.65
	RitualDance	0.52	40.04	0.79	46.23	1.09	53.46	1.53	58.88	2.54	65.40
	Cactus	0.45	34.70	0.73	44.42	1.04	49.99	1.47	57.24	2.58	64.76
	BasketballDrive	0.63	44.99	0.94	50.59	1.26	57.12	1.67	60.13	2.61	66.88
	BQTerrace	0.55	32.36	0.84	41.75	1.11	48.35	1.48	54.42	2.36	62.92
C	BasketballDrill	0.51	23.82	0.97	33.72	1.52	40.31	2.40	45.40	4.44	56.30
	BQMall	0.66	39.34	1.03	46.44	1.40	51.05	1.96	56.37	3.27	63.12
	PartyScene	0.27	35.90	0.52	41.74	0.77	48.33	1.15	53.66	2.11	61.16
	RaceHorsesC	0.34	32.63	0.53	41.36	0.75	46.95	1.07	52.53	1.97	61.01
D	BasketballPass	0.57	38.47	0.86	44.32	1.32	49.27	1.86	53.74	3.19	59.90
	BQSquare	0.21	26.51	0.39	35.61	0.57	40.66	0.85	47.05	1.58	56.36
	BlowingBubbles	0.27	29.87	0.52	38.14	0.82	43.71	1.19	48.98	2.27	57.15
	RaceHorses	0.28	28.52	0.45	37.35	0.72	45.04	1.06	49.90	2.08	56.86
E	FourPeople	0.83	40.33	1.24	47.99	1.71	54.11	2.38	59.09	4.04	66.54
	Johnny	0.97	43.92	1.27	49.98	1.65	55.30	2.13	59.37	3.45	65.65
	KristenAndSara	0.64	41.89	0.97	48.32	1.26	52.96	1.69	56.88	2.86	63.13
Average		**0.46**	**35.22**	**0.72**	**42.98**	**1.01**	**48.80**	**1.42**	**54.20**	**2.43**	**61.34**
Standard deviation (σ)		**0.20**	**6.00**	**0.28**	**5.12**	**0.36**	**5.12**	**0.50**	**4.65**	**0.85**	**4.36**

TABLE VII
COMPARISONS WITH RELATED WORKS

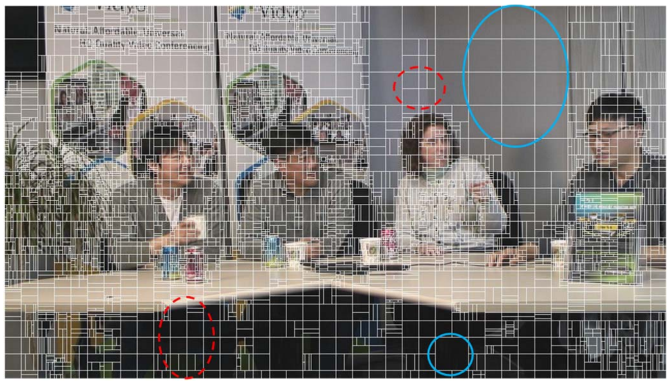
Class	Fu [24]		Yang [25]		Chen [26]		Li [30]		This work			
									C3		*C4*	
	BDBR (%)	ETS (%)	BDBR (%)	ETS (%)	BDBR (%)	ETS (%)	BDBR (%)	ETS (%)	BDBR (%)	ETS (%)	BDBR (%)	ETS (%)
A1	1.31	51.00	0.85	51.39	1.17	42.71	1.60	43.90	0.74	47.82	1.05	53.39
A2	1.19	47.67	0.77	53.08	1.60	48.36	1.49	45.48	0.82	46.38	1.11	52.63
B	0.92	47.60	2.09	58.91	1.56	51.96	1.15	49.09	1.02	52.68	1.40	58.26
C	0.98	42.25	1.48	49.39	1.63	53.79	1.09	45.18	1.11	46.66	1.65	51.99
D	0.62	40.75	1.19	44.16	1.30	53.86	1.07	43.03	0.86	44.67	1.24	49.92
E	1.31	40.00	2.85	58.60	2.55	54.96	1.81	49.50	1.54	54.12	2.07	58.44
Avg	1.02	45.00	1.52	52.01	1.62	51.23	1.32	46.13	1.01	48.80	1.42	54.20
σ	0.41	4.94	0.85	6.47	0.58	6.27	0.45	3.87	0.36	5.12	0.50	4.65

The five operation points of our configurable solution are identified in Fig. 10 as C_1 to C_5. This figure also presents a dotted line showing an extrapolation of our results if using other operation points with different threshold values.

Fig. 10 clearly shows that our solution surpasses all related works since the results of the proposed solution achieved a better tradeoff between ETS and BDBR. This figure also clarifies the high level of flexibility provided by our configurable method compared to the related works since different relations between ETS and BDBR can be explored according to the application requirements.

Table VII presents a more detailed comparison with some of these related works, where average BDBR and ETS results for each video class are presented. For simplicity, considering only two operation points of our configurable solution: C_3 and C_4. These operation points were selected because they are the most comparable with the related works. The related works [24]–[26], and [30] were used in this comparison since they provide detailed results and used almost the same experimental setup considered in our work, making this comparison fairer.

When comparing our work with the solution of Fu *et al.* [24], one can observe that our operation point C_3


(a) Baseline

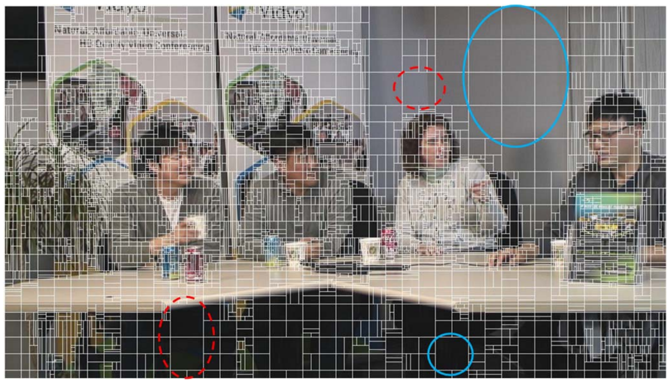

(b) Proposed solution

Fig. 11. Example of CU size distribution obtained with the (a) baseline and (b) proposed solution.

reached a highest ETS (48.80% against 45%) and a little bit smaller BDBR (1.01% against 1.02%). Our work reached similar standard deviation results for BDBR and ETS when compared with [24].

Our operation point C_4, when compared with the work of Yang *et al.* [25], reached a better ETS (54.20% against 52.01%) with a lower BDBR (1.42% against 1.52%). Our work also reached smaller standard deviation results for both ETS and BDBR when compared with [25].

The work of Chen *et al.* [26] has similar results to those presented in [25]. Again, our solution reached a better ETS (54.20% against 51.23%) with a lower BDBR (1.42% against 1.62%). Once more, our work reached a better standard deviation for both ETS and BDBR when compared with [26].

Finally, when comparing our operation point C_3 with Li *et al.* [30], our work reached a better ETS (48.80% against 46.13%) with a lower BDBR (1.01% against 1.32%). When considering the standard deviation, our work reached a better result in BDBR and a worst result in ETS when compared with [30].

As a visual example of the impacts of our method, Fig. 11 presents the CU size distribution obtained with VTM without modifications (baseline) and the proposed solution using the C_3 operation point, considering the first frame of the FourPeople video sequence encoded with AI configuration and QP 32. We choose to analyze the CU size distribution of this video sequence since it presented the highest BDBR impact among the evaluated video sequences using the proposed method. Highlighted regions with a solid blue line are examples of areas where the CUs achieved the same sizes with the original method and with our method. The regions with a dotted red line are examples of areas where the original method and our method reached different CUs sizes. One can notice that even the proposed solution significantly reducing the number of evaluated CU split types, the final distribution of CUs has a similar behavior compared to the baseline VTM encoder, explaining why our solution can provide significant ETS with low impacts on the coding efficiency.

VII. Conclusion

This paper presented a configurable and fast CU partitioning decision using a Light Gradient Boosting Machine (LGBM) to reduce the VVC intra coding time. We trained five LGBM classifiers offline to avoid the evaluation of split types with a high probability of being skipped. The use of effective features extracted from the information of texture, coding, and coding context jointly with a powerful machine learning model allowed the construction of a robust configurable solution capable of efficiently dealing with different video characteristics and resolutions, supporting different application requirements.

The experimental results using five operation points demonstrated that the proposed solution could save from 35.22% to 61.34% of the encoding time at a cost from 0.46% to 2.43% in BDBR. The results also showed that the proposed solution outperforms the related works in terms of the tradeoff between encoding time reduction and coding efficiency. Our method was also the one with the highest time savings among all related works when running at the C_5 configuration (61% against 52% of the best work in the literature). On the other hand, the proposed method when running at the C_1 configuration reached the best BDBR results among the related works (0.5% against 1% of the best work in the literature).

Besides, the high flexibility allowed by our configurable method allows its use in a broad range of applications. The five operation points evaluated in this work can be easily increased and adapted to support other relations between time savings and coding efficiency. Then, one can conclude that this is a powerful solution for VVC encoders targeting real-time applications for different encoding scenarios.

References

[1] S. Dhapola. *COVID-19 Impact: Streaming Services to Dial Down Quality as Internet Speeds Fall*. Indian Express. Accessed: Oct. 2020. [Online]. Available: https://indianexpress.com/article/technology/tech-news-technology/coronavirus-internet-speeds-slow-netflix-hotstar-amazon-prime-youtube-reduce-streaming-quality-6331237/

[2] D. Marpe, T. Wiegand, and G. J. Sullivan, "The H.264/MPEG4 advanced video coding standard and its applications," *IEEE Commun. Mag.*, vol. 44, no. 8, pp. 134–143, Aug. 2006, doi: 10.1109/MCOM.2006.1678121.

[3] G. J. Sullivan, J.-R. Ohm, W.-J. Han, and T. Wiegand, "Overview of the high efficiency video coding (HEVC) standard," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 22, no. 12, pp. 1649–1668, Dec. 2012, doi: 10.1109/TCSVT.2012.2221191.

[4] ITU-T and ISO/IEC, *Versatile Video Coding*, document ITU-T Rec. H.266 and O/IEC 23090-3, 2020.

[5] Y.-W. Huang *et al.*, "A VVC proposal with quaternary tree plus binary-ternary tree coding block structure and advanced coding techniques," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 30, no. 5, pp. 1311–1325, May 2020, doi: 10.1109/TCSVT.2019.2945048.

[6] K. Zhang, Y.-W. Chen, L. Zhang, W.-J. Chien, and M. Karczewicz, "An improved framework of affine motion compensation in video coding," *IEEE Trans. Image Process.*, vol. 28, no. 3, pp. 1456–1469, Mar. 2019, doi: 10.1109/TIP.2018.2877355.

[7] X. Zhao, J. Chen, M. Karczewicz, L. Zhang, X. Li, and W.-J. Chien, "Enhanced multiple transform for video coding," in *Proc. Data Compress. Conf. (DCC)*, Snowbird, UT, USA, Mar. 2016, pp. 73–82, doi: 10.1109/DCC.2016.9.

[8] M. Koo, M. Salehifar, J. Lim, and S.-H. Kim, "Low frequency non-separable transform (LFNST)," in *Proc. Picture Coding Symp. (PCS)*, Ningbo, China, Nov. 2019, pp. 1–5, doi: 10.1109/PCS48520.2019.8954507.

[9] J. Chen, Y. Ye, and S. Kim, *Algorithm Description for Versatile Video Coding and Test Model 10 (VTM 10)*, JVET 19th Meeting, Teleconference, document JVET-S2002, Jul. 2020.

[10] B. Bross *et al.*, *CE3: Multiple Reference Line Intra Prediction (Test 1.1.1, 1.1.2, 1.1.3 and 1.1.4)*, JVET 12th Meeting, document JVET-L0283, Macao, Oct. 2018.

[11] L. Zhao *et al.*, "Wide angular intra prediction for versatile video coding," in *Proc. Data Compress. Conf. (DCC)*, Snowbird, UT, USA, Mar. 2019, pp. 53–62, doi: 10.1109/DCC.2019.00013.

[12] M. Schafer *et al.*, "An affine-linear intra prediction with complexity constraints," in *Proc. IEEE Int. Conf. Image Process. (ICIP)*, Taipei, Taiwan, Sep. 2019, pp. 1089–1093, doi: 10.1109/ICIP.2019.8803724.

[13] S. De-Luxan-Hernandez *et al.*, "An intra subpartition coding mode for VVC," in *Proc. IEEE Int. Conf. Image Process. (ICIP)*, Taipei, Taiwan, Sep. 2019, pp. 1203–1207, doi: 10.1109/ICIP.2019.8803777.

[14] F. Bossen, X. Li, K. Suehring, K. Sharman, V. Seregin, and A. Tourapis, *AHG Report: Test Model Software Development (AHG3)*, JVET 22nd Meeting, Teleconference, document JVET-V0003, Apr. 2021.

[15] *VVC Test Model (VTM)*. Accessed: May 2021. [Online]. Available: https://vcgit.hhi.fraunhofer.de/et/VVCSoftware_VTM

[16] G. Ke *et al.*, "LightGBM: A highly efficient gradient boosting decision tree," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 30, 2017, pp. 3146–3154.

[17] A. Natekin and A. Knoll, "Gradient boosting machines, a tutorial," *Frontiers Neurorobot.*, vol. 7, p. 21, Dec. 2013.

[18] M. Saldanha, G. Sanchez, C. Marcon, and L. Agostini, "Complexity analysis of VVC intra coding," in *Proc. IEEE Int. Conf. Image Process. (ICIP)*, Abu Dhabi, United Arab Emirates, Oct. 2020, pp. 3119–3123, doi: 10.1109/ICIP40778.2020.9190970.

[19] A. Tissier, A. Mercat, T. Amestoy, W. Hamidouche, J. Vanne, and D. Menard, "Complexity reduction opportunities in the future VVC intra encoder," in *Proc. IEEE 21st Int. Workshop Multimedia Signal Process. (MMSP)*, Kuala Lumpur, Malaysia, Sep. 2019, pp. 1–6, doi: 10.1109/MMSP.2019.8901754.

[20] M. Lei, F. Luo, X. Zhang, S. Wang, and S. Ma, "Look-ahead prediction based coding unit size pruning for VVC intra coding," in *Proc. IEEE Int. Conf. Image Process. (ICIP)*, Sep. 2019, pp. 4120–4124, doi: 10.1109/ICIP.2019.8803421.

[21] J. Cui, T. Zhang, C. Gu, X. Zhang, and S. Ma, "Gradient-based early termination of CU partition in VVC intra coding," in *Proc. Data Compress. Conf. (DCC)*, Mar. 2020, pp. 103–112, doi: 10.1109/DCC47342.2020.00018.

[22] Y. Fan, J. Chen, H. Sun, J. Katto, and M. Jing, "A fast QTMT partition decision strategy for VVC intra prediction," *IEEE Access*, vol. 8, pp. 107900–107911, 2020, doi: 10.1109/ACCESS.2020.3000565.

[23] Y. Li, G. Yang, Y. Song, H. Zhang, X. Ding, and D. Zhang, "Early intra CU size decision for versatile video coding based on a tunable decision model," *IEEE Trans. Broadcast.*, early access, Apr. 21, 2021, doi: 10.1109/TBC.2021.3073556.

[24] T. Fu, H. Zhang, F. Mu, and H. Chen, "Fast CU partitioning algorithm for H.266/VVC intra-frame coding," in *Proc. IEEE Int. Conf. Multimedia Expo (ICME)*, Shanghai, China, Jul. 2019, pp. 55–60, doi: 10.1109/ICME.2019.00018.

[25] H. Yang, L. Shen, X. Dong, Q. Ding, P. An, and G. Jiang, "Low-complexity CTU partition structure decision and fast intra mode decision for versatile video coding," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 30, no. 6, pp. 1668–1682, Jun. 2020, doi: 10.1109/TCSVT.2019.2904198.

[26] F. Chen, Y. Ren, Z. Peng, G. Jiang, and X. Cui, "A fast CU size decision algorithm for VVC intra prediction based on support vector machine," *Multimedia Tools Appl.*, vol. 79, nos. 37–38, pp. 27923–27939, Oct. 2020.

[27] T. Amestoy, A. Mercat, W. Hamidouche, D. Menard, and C. Bergeron, "Tunable VVC frame partitioning based on lightweight machine learning," *IEEE Trans. Image Process.*, vol. 29, pp. 1313–1328, 2020, doi: 10.1109/TIP.2019.2938670.

[28] A. Tissier, W. Hamidouche, J. Vanne, F. Galpin, and D. Menard, "CNN oriented complexity reduction of VVC intra encoder," in *Proc. IEEE Int. Conf. Image Process. (ICIP)*, Oct. 2020, pp. 3139–3143, doi: 10.1109/ICIP40778.2020.9190797.

[29] J. Zhao, Y. Wang, and Q. Zhang, "Adaptive CU split decision based on deep learning and multifeature fusion for H.266/VVC," *Sci. Program.*, vol. 2020, Aug. 2020, Art. no. 8883214.

[30] T. Li, M. Xu, R. Tang, Y. Chen, and Q. Xing, "DeepQTMT: A deep learning approach for fast QTMT-based CU partition of intra-mode VVC," *IEEE Trans. Image Process.*, vol. 30, pp. 5377–5390, 2021, doi: 10.1109/TIP.2021.3083447.

[31] S.-H. Park and J. Kang, "Fast multi-type tree partitioning for versatile video coding using a lightweight neural network," *IEEE Trans. Multimedia*, early access, Dec. 2, 2020, doi: 10.1109/TMM.2020.3042062.

[32] F. Bossen, K. Suhring, A. Wieckowski, and S. Liu, "VVC complexity and software implementation analysis," *IEEE Trans. Circuits Syst. Video Technol.*, early access, Apr. 9, 2021, doi: 10.1109/TCSVT.2021.3072204.

[33] Y.-K. Wang *et al.*, "The high-level syntax of the versatile video coding (VVC) standard," *IEEE Trans. Circuits Syst. Video Technol.*, early access, Apr. 5, 2021, doi: 10.1109/TCSVT.2021.3070860.

[34] H. Schwarz *et al.*, "Quantization and entropy coding in the versatile video coding (VVC) standard," *IEEE Trans. Circuits Syst. Video Technol.*, early access, Apr. 9, 2021, doi: 10.1109/TCSVT.2021.3072202.

[35] J. Pfaff *et al.*, "Intra prediction and mode coding in VVC," *IEEE Trans. Circuits Syst. Video Technol.*, early access, Apr. 12, 2021, doi: 10.1109/TCSVT.2021.3072430.

[36] M. Karczewicz *et al.*, "VVC in-loop filters," *IEEE Trans. Circuits Syst. Video Technol.*, early access, Apr. 9, 2021, doi: 10.1109/TCSVT.2021.3072297.

[37] H. Gao, X. Chen, S. Esenlik, J. Chen, and E. Steinbach, "Decoder-side motion vector refinement in VVC: Algorithm and hardware implementation considerations," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 31, no. 8, pp. 3197–3211, Aug. 2021, doi: 10.1109/TCSVT.2020.3037024.

[38] A. Kammoun *et al.*, "Forward-inverse 2D hardware implementation of approximate transform core for the VVC standard," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 30, no. 11, pp. 4340–4354, Nov. 2020, doi: 10.1109/TCSVT.2019.2954749.

[39] Y. Fan, Y. Zeng, H. Sun, J. Katto, and X. Zeng, "A pipelined 2D transform architecture supporting mixed block sizes for the VVC standard," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 30, no. 9, pp. 3289–3295, Sep. 2020, doi: 10.1109/TCSVT.2019.2934752.

[40] L. Zhu, Y. Zhang, S. Wang, S. Kwong, X. Jin, and Y. Qiao, "Deep learning-based chroma prediction for intra versatile video coding," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 31, no. 8, pp. 3168–3181, Aug. 2021, doi: 10.1109/TCSVT.2020.3035356.

[41] Z. Zhang *et al.*, "Fast DST-VII/DCT-VIII with dual implementation support for versatile video coding," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 31, no. 1, pp. 355–371, Jan. 2021, doi: 10.1109/TCSVT.2020.2977118.

[42] F. Bossen, J. Boyce, X. Suehring, X. Li, and V. Seregin, *JVET Common Test Conditions and Software Reference Configurations for SDR Video*, JVET 14th Meeting, document JVET-N1010, Mar. 2019.

[43] G. Bjontegaard, *Calculation of Average PSNR Differences Between RDCurves*, VCEG Meeting, document VCEG-M33, 2001, pp. 1–5.

[44] K. Sharman and K. Suehring, *Common Test Conditions*, JCT-VC 26th Meeting, document JCTVC-Z1100, Jan. 2017.

[45] T. Daede, A. Norkin, and I. Brailovskiy, *Video Codec Testing and Quality Measurement*, document draft-IETF-NETVC-testing-08 (work in progress), 2019, p. 23.

[46] *Video Test Media [Derf's Collection]*. Accessed: May 2021. [Online]. Available: https://media.xiph.org/video/derf/

[47] *Feature Selector*. Accessed: May 2021. [Online]. Available: https://github.com/WillKoehrsen/feature-selector

[48] T. Akiba, S. Sano, T. Yanase, T. Ohta, and M. Koyama, "Optuna: A next-generation hyperparameter optimization framework," in *Proc. 25th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, 2019, pp. 2623–2631.

[49] J. Bergstra, R. Bardenet, Y. Bengio, and B. Kégl, "Algorithms for hyper-parameter optimization," in *Proc. Adv. Neural Inf. Process. Syst. Found. (NIPS)*, vol. 24, 2011, pp. 2546–2554.

Mário Saldanha received the B.S. and M.Sc. degrees in computer science from the Federal University of Pelotas (UFPel), Pelotas, Rio Grande do Sul, Brazil, in 2016 and 2018, respectively, where he is currently pursuing the Ph.D. degree. He is also a member of the Video Technology Research Group (ViTech), UFPel. His research interests include complexity reduction and hardware-friendly algorithms for 2D/3D video coding.

Gustavo Sanchez (Member, IEEE) received the B.S. degree in computer science from the Federal University of Pelotas (UFPel) in 2012, the degree in electrical engineering from Sul-Rio-Grandense Federal Institute of Education, Science and Technology in 2013, the M.Sc. degree in computer science from UFPel in 2014, and the Ph.D. degree in computer science from the Pontifical Catholic University of Rio Grande do Sul in 2019. He has been a Professor with IFFarroupilha, Brazil, since 2014. He has more than ten years of research experience on algorithms and hardware architectures for video coding. His research interests include complexity reduction algorithms, hardware-friendly algorithms, and dedicated hardware design for 2D/3D video coding.

César Marcon (Senior Member, IEEE) received the Ph.D. degree in computer science from the Federal University of Rio Grande do Sul, Brazil, in 2005. He has been a Professor with the Polytechnic School of the Pontifical Catholic University of Rio Grande do Sul (PUCRS), Brazil, since 1995. He is a Distinguished Brazilian Researcher with the CNPq PQ-2 Grant. Since 2005, he coordinated several research projects in the areas of telecom and healthcare. He is an Advisor of M.Sc. and Ph.D. graduate students at the Graduate Program in Computer Science (PPGCC), PUCRS. He has more than 150 articles published in prestigious journals and conference proceedings. His research interests are embedded systems in the telecom domain, MPSoC architectures, partitioning and mapping application tasks, fault tolerance, and real-time operating systems. He is a Senior Member of the Brazilian Computer Society (SBC).

Luciano Agostini (Senior Member, IEEE) received the M.S. and Ph.D. degrees from the Federal University of Rio Grande do Sul, Porto Alegre, Brazil, in 2002 and 2007, respectively. He was the Executive Vice President for Research and Graduate Studies of the Federal University of Pelotas (UFPel), Brazil, from 2013 to 2017. Since 2002, he has been a Professor at UFPel, where he leads the Video Technology Research Group (ViTech). He is currently a Brazilian Distinguished Researcher through the CNPq PQ-1D Grant. He is also an Advisor at UFPel master's and Ph.D. in computer science courses. He has more than 300 published articles in journals and conference proceedings. His research interests include 2D and 3D video coding, algorithmic optimization, arithmetic circuits and digital systems. He is a Senior Member of ACM. He is a member of the IEEE CAS, CS, and SPS societies. At the IEEE CAS, he is a member of the MSATC. He is also a member of SBC and SBMicro Brazilian societies. He is an Associate Editor of two IEEE CAS journals, such as IEEE TRANSACTIONS ON CIRCUITS AND SYSTEMS FOR VIDEO TECHNOLOGY and IEEE OPEN JOURNAL OF CIRCUITS AND SYSTEMS.