Fast 3D-HEVC Depth Map Encoding Using Machine Learning

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

***Abstract*— This paper presents a fast depth map encoding for 3D-High Efficiency Video Coding (3D-HEVC) based on static decision trees. We used data mining and machine learning to correlate the encoder context attributes, building the static decision trees. Each decision tree defines that a depth map Coding Unit (CU) must be or not be split into smaller blocks, considering the encoding context through the evaluation of the encoder attributes. Specialized decision trees for I-frames, P-frames and B-frames define the partitioning of 64 × 64, 32 × 32, and 16 × 16 CUs. We trained the decision trees using data extracted from the 3D-HEVC Test Model considering all-intra and random-access configurations, and we evaluated the proposed approach considering the common test conditions. The experimental results demonstrated that this approach can halve the 3D-HEVC encoder computational effort with less than 0.24% of BD-rate increase on the average for all-intra configuration. When running on random-access configuration, our solution is able to reduce up to 58% the complete 3D-HEVC encoder computational effort with a BD-rate drop of only 0.13%. These results surpass all related works regarding computational effort reduction and BD-rate.**

***Index Terms*— 3D-HEVC, depth maps, machine learning, decision trees, time saving.**

I. Introduction

THREE-DIMENSIONAL (3D) video systems have evolved considerably in the last few years due to their real-world visual experience that goes beyond Two-Dimensional (2D) videos. 3D videos allow the users to enjoy an experience with depth perception by employing techniques such as stereoscopic, multi-view and Multi-View plus Depth (MVD), being applied in multimedia applications such as 3D movies, Free-Viewpoint TV [1], autostereoscopic systems and others.

The experts of the Joint Collaborative Team on Video Coding (JCT-VC) developed the High-Efficiency Video Coding (HEVC) standard [2] for reaching high compression rates on 2D high definition videos. Based on HEVC, the Joint Collaborative Team on 3D Video Coding Extension Development (JCT-3V) drawn the 3D-HEVC standard [3], [4] to perform an advanced 3D video coding intending to reach high compression efficiency.

There are some recent works [5]–[8] focusing on developing fast encoding solutions for 2D videos. The 3D video encoding is even a more complex system, requiring the encoding of multiple views at the same time, demanding more bandwidth for video transmission and a larger amount of space for video storage. The amount of data increases proportionally with the number of cameras/views used to capture scenarios for simulcast or multi-view coding. 3D-HEVC adopts the MVD data format to increase the encoding efficiency, which is an alternative to multi-view video format that considers a set of texture views and the associated depth maps. Lightweight synthesis techniques, such as Depth Image Based Rendering (DIBR) [9], are applied to interpolate the texture views based on the depth maps, producing as many synthesized or virtual views as required at decoder/receiver side [9]. Thus, only a subset of views (texture and depth) is encoded/transmitted, and the decoder/receiver side can generate virtual views without needing to encode/transmit intermediary views.

A depth map describes geometrical information of the scene indicating the distance between a camera and objects using gray shades [3]. The characteristics of depth maps are very different from texture frames since depth maps have large homogeneous regions delimited by sharp edges, whereas texture frames contain a complex content with sudden variations in the sample values [10]. Although depth maps are not presented to viewers [11], the geometrical information is crucial for generating the synthesized views. During the encoding, distortions on the edge information of the depth maps can cause inaccurate representation between foreground and background pixels in the process of synthesizing views, changing the structure of the 3D videos. Therefore, the preservation of edge information is an essential task when encoding depth maps to provide high quality synthesized views [12]. In homogeneous regions, the problem is reduced because human eyes cannot perceive small changes in the depth values [13].

The 3D-HEVC depth map coding introduces some new tools such as Depth Modeling Modes (DMM) [14], Depth

Manuscript received July 21, 2018; revised October 21, 2018 and December 27, 2018; accepted January 28, 2019. Date of publication February 7, 2019; date of current version March 5, 2020. This paper was performed in cooperation with Hewlett-Packard Brazil Ltda., using incentives of Brazilian Informatics Law (Law no 8.248 of 1991). This work was financed in part by the Coordenação de Aperfeiçoamento de Pessoal de Nível Superior Brasil (CAPES), Brazil-Finance Code 001-, and also by the Brazilian research support agencies CNPq and FAPERGS. This paper was recommended by Associate Editor W. Liu. *(Corresponding author: Mário Saldanha.)*

M. Saldanha and L. Agostini are with the Graduate Program in Computer Science (PPGC), 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).

G. Sanchez is with the IF Farroupilha, Alegrete 97555-000, Brazil, and also with the Graduate Program in Computer Science, Pontifical Catholic University of Rio Grande do Sul, Porto Alegre 90619-900, Brazil (e-mail: gustavo.sanchez@acad.pucrs.br).

C. Marcon is with the Graduate Program in Computer Science, 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 of the figures in this article are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TCSVT.2019.2898122

Intra Skip [15], and Segment-wise Direct Component Coding (SDC) [16] targeting an efficient encoding. Besides, depth maps have the same flexible quadtree-based structure that HEVC uses for texture frames [2]. In this structure, each frame is divided into *Coding Tree Units* (CTUs), and each CTU can be recursively divided into smaller blocks, called *Coding Units* (CUs). Finally, each CU can be split into two or four *Prediction Units* (PUs), which are individually evaluated. In the 3D-HEVC Test Model (3D-HTM) [17], a Rate-Distortion Optimization (RDO) process is performed at each level of the quadtree to determine the best coding mode and the best partition size for a CU at that level.

The RDO evaluation is a complex task, requiring a high computational effort. To minimize this effort, several works [22]–[32] have recently proposed techniques to simplify the 3D-HEVC depth map coding and partitioning structure. Although these works reduce the encoding time and complexity, most of them significantly reduce the encoding efficiency.

The use of machine learning in video coding applications is a promising approach to reduce the encoder computational effort, minimizing the Rate-Distortion (RD) efficiency losses. Such solution was applied in some recent works targeting different video coding domains, like the conventional HEVC coding [33]–[35], the HEVC Screen Content Coding extension [36], and the H.264/AVC to HEVC transcoding [37].

Our previous work [38] proposed a quadtree limitation for the depth map intra-prediction to reduce the computational complexity and minimize the RD efficiency losses. This quadtree limitation approach uses *Data Mining* (DM) as a tool to build a set of decision trees for finding the best size for a current CU; consequently, allowing terminate the quadtree decision process early. The structure decision of the CU partitioning is seen as a data classification problem, which is efficiently solved by a set of tests.

This article extends our previous work [38], which covered only intra-frame prediction, targeting a complete fast 3D-HEVC depth map encoding considering intra- and inter-frame predictions. This solution builds decision trees employing machine learning to define the quadtree splitting; thus, avoiding the full RDO calculation. Decision trees were selected to be used in this investigation due to its simplicity and high accuracy for this kind of problem. Besides, decision trees have a hardware-friendly characteristic, allowing an easier and efficient future hardware design. However, other machine learning models will be explored in future works of our research group targeting video encoding optimizations.

Since intra-frames (I-frames) are encoded with different partitioning structures and coding tools than the unidirectional frames (P-frames) or bi-directional frames (B-frames) [2], we built two new sets of trees: one for I-frames and other for P- and B-frames. Each set encompasses specialized decision trees for each CU size (16×16, 32×32 and 64×64), totalizing six decision trees. This solution significantly reduces the depth map encoding effort with negligible encoding efficiency degradation. To the best of authors' knowledge, this is the first work in the literature (besides our previous work [38]) using decision trees for reducing the 3D-HEVC depth map encoding effort.

The remaining of this paper is divided as follows. Section II discusses related works focusing on the computational effort reduction for the 3D-HEVC coding. Section III presents the 3D-HEVC coding structure including the quadtree structure and I-, P-, and B-frames. Section IV provides an initial analysis to motivate the proposed solution. Section V details the proposed solution. Section VI shows and discusses the experimental results. Finally, Section VII renders the conclusions of this work.

II. Related Work

Although several works, such as [18]–[32], focus on saving time on depth map coding, only a few of them ([18], [21]–[25]) emphasis on performing this timesaving using CU quadtree depth limitation techniques. The works [22] and [23] focus on limiting only the I-frames CU quadtrees, while [21], [24], and [25] are applied in P- and B-frames to limit the CU quadtree depth. Moreover, some works (e.g., [54]) perform a CU quadtree limitation in 3D-HEVC; however, this technique is applied only in texture coding. The technique exploited in [54] predicts the encoding CU level in the dependent texture views using CU level information of neighboring views already encoded. Thus, Subsections II-A and II-B present related solutions focusing on intra-frame (I-frame) and inter-frame (P- and B-frames) complexity reduction, respectively.

A. Intra-Frame Prediction Solutions

Shen *et al.* [18] classified the encoding CU in three types (simple, normal and complex) using the encoding mode and the RD-cost of previous encoded CUs in intra prediction. When the encoding CU is classified as simple, then only planar and DC modes are evaluated. If CU is classified as normal, then planar, DC and angular modes are evaluated. Finally, when the CU is classified as complex, all those modes and DMMs are evaluated. Besides, Shen *et al.* [18] also propose to limit the quadtree expansion by checking the current RD-cost and verifying if the best mode selected at intra-frame prediction is DC or planar.

Zhang *et al.* [19] designed a fast depth intra mode decision using only planar, DC, vertical and horizontal modes in HEVC intra-frame prediction. DMMs are only executed when the best encoding mode in HEVC intra-frame prediction is planar or DC. Besides, the DMM-1 encoding effort is reduced by evaluating only a subset of their wedgelets.

Chen *et al.* [20] created a technique for detecting edge regions in depth map coding. If the encoding block does not contain an edge, then the DMMs are not evaluated, and the quadtree expansion is finished.

Peng *et al.* [22] proposed an approach that mixes block- and quadtree-level decision algorithms. The block-level decision uses a threshold based on the RD-cost of the prediction modes. The quadtree-level algorithm computes the variance of the CU and the maximum variance of the sub-blocks. The split only occurs if the maximum variance of sub-blocks is higher than the CU variance or if the CU variance is higher than a threshold.

Zhang *et al.* [23] exploit a QP-based quadtree depth limit to detect if the information of smaller blocks is relevant or their evaluation can be skipped. This solution is based on regions of interest (edge regions), which are extracted by applying the Corner Point (CP) [55] feature. CPs indicate the details level of each block to predict the quadtree for the current CU. Thus, the evaluation of smaller sizes of CUs is avoided when regions without significant information are computed. Whereas, the evaluation of larger sizes of CUs can be skipped in blocks containing significant details.

B. Inter-Frame Prediction Solutions

Zhang *et al.* [21] speed up the depth map encoding by correlating the depth maps with the texture information and classifying edges. The texture information is used to predict the depth level of the current depth map CU and to decide for evaluating only *Merge/Skip* mode (if all texture predictors are encoded with *Merge/Skip*). Whereas, they use the Canny operator algorithm to classify a depth map CU as an edge region or no edge region. When a depth map CU is classified as no edge region, the depth map CU is evaluated only with *Merge/Skip*, inter 2N $\times$ 2N, planar and DC modes.

Mora *et al.* [24] propose an inter-component tool called Quadtree Limitation (QTL) that limits the quadtrees for P- and B-frames (QTL is not applicable to I-frames). Their proposal is based on the idea a texture frame and its associated depth map represent the same scene at the same time and viewpoint; therefore, their quadtree structure is closely linked. Thus, the QTL tool is used to predict the depth level of the depth map quadtree based on the texture quadtree decision; i.e., the depth map quadtree is constrained to the same level of the texture quadtree. The current version of the 3D-HTM reference software adopts the QTL tool as a default technique.

Lei *et al.* [25] propose a fast mode decision algorithm for depth map coding based on the grayscale similarities and the inter-view correlation, which is composed of early CU termination and early PU mode decision for encoding dependent views. The CU termination decision uses a threshold for classifying the grayscale similarities between the current encoding CU and the co-located block in the reference frame. When there is a high similarity, the quadtree depth level of the current CU is constrained by the level of the co-located quadtree in the reference frame. A grayscale similarity and an inter-view correlation are applied in the PU mode decision, which finalizes when the co-located CU in reference view selects *Merge* or *Inter* 2N $\times$ 2N as best mode. Besides, different strategies are proposed to encode P- and B-frames of dependent views. P-frames also use spatial information of adjacent CUs for early PU decision.

This article presents the first work using data mining and machine learning to evaluate the complete encoder attributes of the 3D-HEVC depth maps and extract the correlations of these attributes, avoiding some dynamic encoder decisions inside intra and inter predictions, reducing the encoder complexity. Liu *et al.* [33] and Correa *et al.* [34] also used machine learning techniques (convolution neural network and decision trees, respectively) but applied to a 2D encoder (texture only).

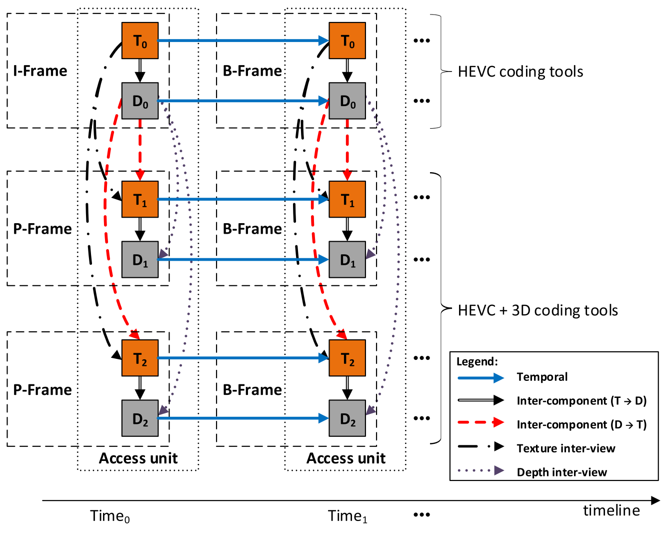


Fig. 1. Basic 3D-HEVC coding structure with temporal, inter-view, and inter-component prediction.

Since the scenario of this work is entirely different, a new and complete evaluation of the encoder attributes correlations was necessary, conducting the definition of new attributes and allowing the definition of inedited static decision trees.

III. 3D-HEVC Coding Structure

The 3D-HEVC coding structure is based on Access Units (AUs), which contain all texture frames and the respective depth maps. To encode each frame in an AU, 3D-HEVC defines an advanced quadtree-based partitioning structure, for both texture and depth data. Fig. 1 shows the basic 3D-HEVC coding structure and the dependencies among AUs, views, components (texture/depth) and I-, P- and B-frames. An AU can contain an arbitrary number of views; the first view to be encoded is known as the **base view**. Fig. 1 relies on features implemented on the 3D-HTM reference software [17], and it illustrates the base view (T_0) and two dependent texture views (T_1, T_2) beyond their correspondent depth maps (D_0, D_1, D_2). T_0 is independent of other views, being encoded by an HEVC encoder since an HEVC decoder provides images for the 2D-conventional display from a bitstream generated by a 3D-HEVC encoder. However, the dependent views are encoded using data from the base view to reducing the inter-view redundancy through a process called Disparity Compensation Prediction (DCP) [39], [40].

3D-HEVC uses a hierarchical B-frame structure; the anchor frame of the base view and their correspondent depth map are I-frames, and the remaining are B-frames, whereas the anchor frames of dependent views are P-frames and the remaining are also B-frames. The encoding tools of the base view according to the frame type (I- and B-frames) are presented next:

- I-frames of depth maps are encoded using the HEVC intra-prediction and new intra-prediction tools, such as DMM-1, DMM-4, SDC and DIS [41]. Although DMM-4 is classified as an intra-prediction tool, it also uses inter-component information from texture frame;

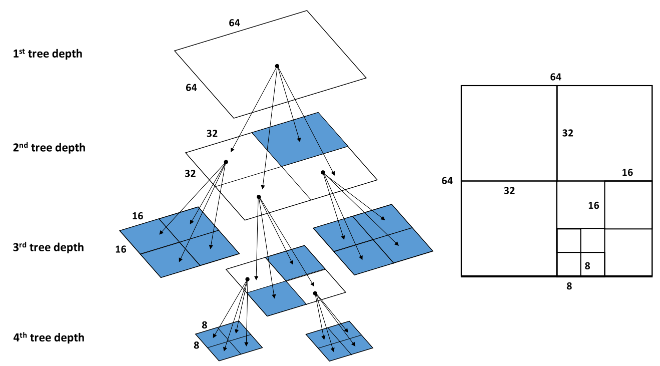


Fig. 2. 3D-HEVC quadtree structure of a CTU split into CUs.

- I-frames of texture are encoded using only the HEVC intra-prediction, then the I-frames are fully compliant with a conventional HEVC codec;
- B-frames of texture and depth maps are encoded with the tools employed in I-frames and, also with the bi-directional Motion Estimation (ME) [42]. Moreover, the *Merge/Skip Mode* (MSM) [43] is evaluated to reduce the temporal redundancy. B-frames of texture base view are also fully compliant with a conventional HEVC codec.

The encoding process of dependent views are presented below, considering the allowed frame types (P- and B-frames):

- The first P-frame of texture is encoded with: (i) the HEVC intra-prediction; (ii) the Disparity Compensation Prediction (DCP); and (iii) the inter-component depth to texture using tools such as Depth Based Block Partitioning (DBBP) [44]. When there are reference frames for texture P-frame, also it is used the unidirectional ME, which can evaluate MSM;
- The first P-frame of depth maps is encoded with the tools used in depth maps I-frames and, also with DCP. The remaining P-frames of depth maps also can be encoded with unidirectional ME;
- B-frames of texture and depth maps are encoded with the same tools of P-frames plus the bi-directional ME tool [42].

3D-HEVC inherits an advanced and flexible quadtree-based structure from HEVC, where a frame is divided into *Coding Tree Units* (CTUs), and each CTU can be recursively split in a quadtree structure where the leaves are *Coding Units* (CUs). Finally, each CU can be divided into two or four *Prediction Units* (PUs), which are separately predicted through the intra- or inter-frame processes [4]. The same process applied in the texture coding is used in the depth map coding.

Fig. 2 exemplifies a 64×64 CTU split into several CUs with different quadtree depth levels, where the tree leaves (filled squares) are the final encoded CUs. Commonly, 3D-HTM sets the CTU size in 64×64 and a CU can be a single 64×64 block (1st tree depth), or the CTU can be recursively split into four blocks until reaching the 8×8 size (4th tree depth).

All PUs use either intra- or inter-frame prediction. The intra-frame prediction uses only the HEVC intra-prediction modes for texture data, whereas, for depth data, DMMs (used mainly in edge regions) and DIS (used when homogeneous regions are encoded) are also available, together with the conventional intra-prediction [41]. The ME, DCP and MSM tools are available for both texture and depth maps in PU inter-frame prediction. MSM is conceptually the same used in HEVC, and it is similar to the *SKIP* mode of H.264/AVC [45]. Besides, 3D-HEVC texture coding uses DBBP, which defines the block partitioning based on the corresponding depth block [44]. 3D-HEVC defines the quadtree structure evaluating the CU sizes, PU partitions and PU modes with an iterative splitting process employing RDO.

IV. Initial Analysis and Motivation

We implemented two initial experiments to evaluate the impact of depth map complexity inside the global 3D-HEVC encoder and, also the encoder behavior regarding Quantization Parameters (QPs). Both experiments evaluate All-Intra (AI) and Random-Access (RA) encoder configurations [47] using the 3D-HTM version 16.0 [17]. Besides, the experiments covered the Common Test Conditions (CTC) [47] using eight sequences and four pairs of Quantization Parameters (QP-pair). The experiments covered the 3D video sequences: *Balloons* [48], *Kendo* [48] and *Newspaper_CC* [49] with a resolution of 1024×768 pixels, and *GT_Fly* [50], *Poznan_Hall2* [51], *Poznan_Street* [51], *Undo_Dancer* [52] and *Shark* [53] with a resolution of 1920×1088 pixels. Each QP-pair defines the pair of quantization parameters used to encode texture and depth maps (QP-texture, QP-depth) [47]. The four QP-pairs evaluated were (25, 34), (30, 39), (35, 42) and (40, 45). Additionally, to have accurate results, we performed these evaluations turning off the quadtree limitation available by the 3D-HTM default, because this default limitation already reduces the encoding effort at the cost of coding efficiency losses.

Fig. 3 shows the distribution of the 3D-HEVC encoding effort considering the average results for all CTC sequences and four different QP-pairs. In AI configuration, the encoder spends an average of 83% and 17% of the time to encode depth maps and texture, respectively, as displayed in Fig. 3(a). It happens in the AI scenario because texture coding only applies the HEVC intra-frame prediction, whereas, depth map coding also employs DMMs, DIS, and SDC evaluations [41]. Considering the AI configuration, the depth map coding is 4.8 times more time demanding than the texture coding, on average.

Fig. 3(b) shows QP-pair has a higher influence for the RA configuration than for the AI configuration. Here, the depth maps still require higher encoding effort than texture. The highest difference is found for the QP-pair (40, 45), where 80% of the processing time is used to encode depth maps and 20% to encode texture. The smallest difference occurs on the QP-pair (25, 34), where texture coding and depth map coding use 31% and 69% of the encoder time, respectively. The depth map encoding concerning RA configuration is 2.9 times more time consuming than the texture coding, on average. This evaluation shows the depth map coding is a bottleneck in the 3D-HEVC encoding and solutions for reducing the computational effort required to encode depth maps without penalizing the encoding efficiency are crucial.

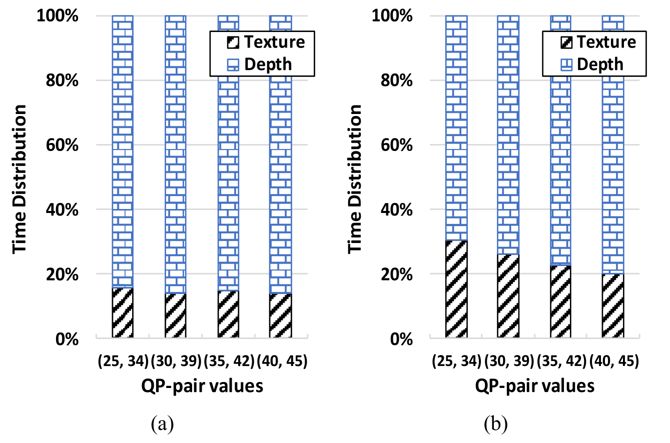


Fig. 3. 3D-HEVC complexity distribution regarding four QP-pairs in (a) AI and (b) RA configurations.

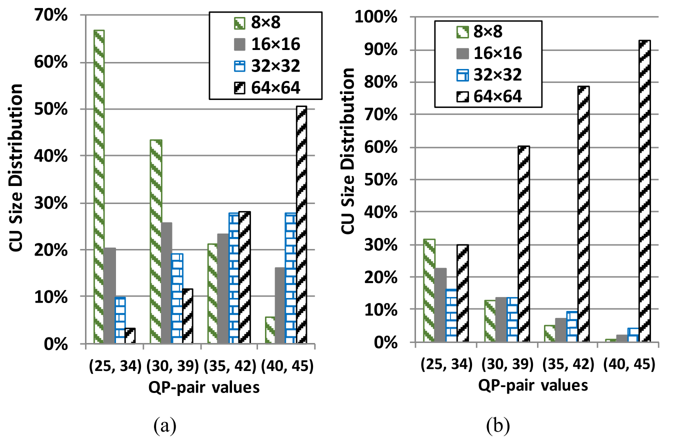


Fig. 4. Distributions of CU sizes for 3D-HEVC depth maps regarding four QP-pairs in (a) AI and (b) RA configurations.

The second experiment evaluated the influence of the encoder attributes and configurations in the internal encoder decisions. As previously mentioned, the 3D-HEVC standard provides a flexible encoding structure; and the implementation of the 3D-HTM encoder employs a complex RDO-based process to assess the combinations of encoding structures (block partitions and prediction modes) that allows choosing the one with the lowest RD-cost. Fig. 4(a) and (b) display the results of CU size distribution for four values of QP-pair considering AI and RA configurations, respectively.

The variation in the QP affects the CU size distribution for both configurations because QP defines the compression rate influencing the image quality. High QPs generate more homogeneous areas in the coded image that are efficiently encoded using bigger CU sizes. However, with a low QP, the predicted images tend to preserve several details, requiring lower CU sizes to maintain the encoding efficiency.

In our analysis, RA configuration uses less small CUs when compared to the AI configuration for all evaluated QP-pairs, because the RA configuration allows the inter-frames and inter-views redundancies exploration using motion and disparity estimation. The exploration of these redundancies in P- and B-frames allows higher encoding efficiency reusing bigger blocks from reference frames.

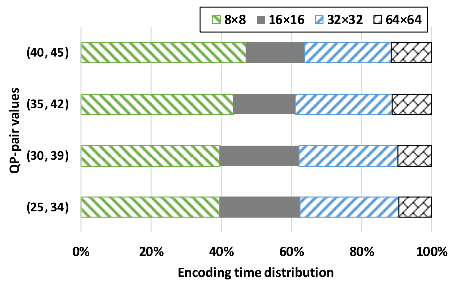


Fig. 5. Encoding time distribution of each CU size for four QP-pairs considering AI configuration.

Fig. 5 details the computational cost to evaluate each CU size, considering only depth map encoding time and AI configuration. One can observe there is a high computational cost associated with the evaluation of smaller CU sizes, such as 8×8. Also, one can conclude the smallest encoding effort is spent in 64×64 CUs for all QP-pair values.

Joining the presented results, one can conclude an early decision algorithm that limits the encoding of smaller CUs could provide a good tradeoff between encoder decision accuracy and high encoding time reduction.

Although the QP values and encoder configurations have a substantial influence on the quadtree definition, the simple removal of some quadtree level is not an appropriate solution for reducing the encoding effort, since this decision can significantly degrade the encoding efficiency (bit-rate versus quality). However, a solution able to decide when a current CU should not be split into smaller CUs considering the QP value, the encoder configurations and other encoder attributes, can avoid the high cost of evaluating the full RDO process. Then, a most robust solution should be provided to reduce complexity with minor impact on the coding efficiency.

The encoder has many other decisions defined at runtime. Then, solutions that statically fulfill some of these decisions considering the encoder context (and not the full RDO) and with less impact on the encoder efficiency are highly desirable.

Decision trees can be used to predict the value of dependent variables identifying regularities and building generalizations in attributes of the data set. These models are built through the data mining techniques, frequently used when high accuracy and low complexity execution are required [46]. Then, decision trees could be an interesting solution to deal with the encoder decisions considering the encoder context and avoiding the full RDO process.

V. Fast CU Encoding Based on Machine Learning

This article presents a fast depth map encoding using machine learning to define decision trees which are used in the encoder CU quadtree split decision process instead of the full RDO process. Section V-A explains the methodology employed to define the decision trees. Sections V.B and V.C explain the developed decision trees for I-frames and P- and B-frames, respectively.

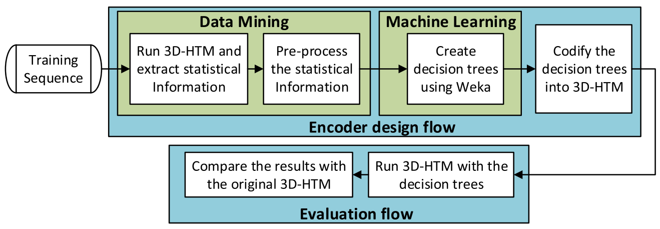

Data Mining
Machine Learning
Training Sequence
Run 3D-HTM and extract statistical Information
Pre-process the statistical Information
Create decision trees using Weka
Codify the decision trees into 3D-HTM
Encoder design flow
Compare the results with the original 3D-HTM
Run 3D-HTM with the decision trees
Evaluation flow

Fig. 6. Methodology for design and evaluation of the encoder with the decision trees.

A. Methodology

This article explores the use of data mining and machine learning based on attributes with high correlation with the CU split decision to build static decision trees to determine when a CU should be split into smaller ones. The split choice occurs when the decision tree classifies the encoding CU as requiring further evaluations; otherwise, the computation of that CU is finalized. The use of data mining and machine learning allows discovering correlations between the encoding context and its attributes. If these correlations are strong enough, it is possible to define static decisions trees to reduce the encoding complexity with negligible impact on the coding efficiency.

Based on this exploration, this work proposes the usage of six static decision trees. One tree is necessary for splitting decisions of each CU size (16×16, 32×32 and 64×64). Besides, I-frames use different encoder tools than P- and B-frames, generating different contexts. Thus, one group of trees is necessary for I-frames and another one for P- and B-frames.

Fig. 6 summarizes the methodology used in this work. The first step is the application of data mining to define the most relevant encoder attributes that must be considered. This step was performed by executing the 3D-HTM and extracting statistical data correlating the CU split decision with the evaluated attributes. Relevant information for the decision process and a flag indicating if each CU has been split or not were stored. The data mining process is responsible for pre-processing the statistical data collected in the previous step. The pre-processing balances the input-data, organizing it in two equal-sized sets containing 50% of entries that lead to the CU split and 50% of entries that do not split the CUs.

The statistical information obtained in the data mining process is used to feed the machine learning process and train the decision trees. The *Waikato Environment for Knowledge Analysis* (WEKA) [56], version 3.8, was used for creating each decision tree. The decision trees were created using the J48 algorithm, which is an open-source implementation of the C4.5 algorithm [57] available on WEKA.

We performed the *Reduced Error Pruning* (REP) algorithm [58] in each tree, to avoid the overfitting problem on the dataset training. This procedure reduced the size of the decision trees, contributing to accelerate the encoding process and contributing to an easier and faster future hardware implementation. Additionally, REP allows a better feature generalization of the depth map CUs at the cost of a small accuracy loss.

TABLE I
CONFIGURATIONS USED IN THE EXPERIMENTS

Profile	AI	RA
GOP size	1	8
Intra-period	1	24
Bits per pixel	8	
Texture/Depth views	3/3	
QP-pair	(25, 34), (30, 39), (35, 42), (40, 45)	
DMM evaluation	Enabled	
Depth intra skip	Enabled	
VSO	Enabled	
Fast Options-QTL	Disabled	
Rate control	Disabled	
3D-HTM version	16.0	

The created decision trees were inserted in 3D-HTM, substituting the conventional RDO process. The final evaluation was done using additional video sequences; i.e., video sequences different from that used in the decision tree definition.

We selected randomly two video sequences from CTC (*Kendo* [48] and *Poznan_Street* [51]) and, *Champagne_tower* [48] and *Pantomime* [48] to be used in the training stage. To demonstrate the robustness of the proposed method, we generated three sets of trees using different training sequences. Firstly, we apply only the *Kendo* sequence in the training process generating three trees for I-frames (the same published in our previous work [38]) and three trees for P- and B-frames. Secondly, we increased the number of sequences used in the training process using *Kendo* and *Poznan_Street* in this step. Thus, other six threes were generated (three for I-frames and three for P- and B-frames). Thirdly, we generate six decision trees using four sequences (*Kendo*, *Poznan_Street*, *Champagne_tower*, and *Pantomime*) for the training process. The reached results are discussed in the next sections and showed that this approach is robust even when only one sequence is used in the training process since the reached results were similar for the three training sets.

Different video sequences were evaluated in Section VI to demonstrate the trained solution can achieve high quality in different encoding scenarios. It is important to emphasize there is a limited number of 3D video sequences with their depth maps available to make 3D video coding experiments.

Table I shows the main configurations applied to the experiments for each encoder setting. These configurations are in accordance with the CTC definitions. However, small changes in the configuration scenario tend to lead to similar results, considering the full quadtree search process is not affected by these changes.

Sections V-B and V-C discuss the generate decision trees using *Kendo* as the training sequence for I-frames and, P- and B-frames, respectively.

B. Intra-Frame Decision Trees

We collected a large amount of data from the depth video sequences and internal encoding variables to find features that could lead to effective decisions of CU splitting. The all-intra (AI) configuration was considered to define the intra-frame decision trees since this configuration only allows the use of intra prediction tools. The attributes listed in Table II

TABLE II
ATTRIBUTES EVALUATED IN I-FRAMES

Attribute	Description
Average	Average value of original samples of the current CU
VAR	Variance of the CU original samples
VAR_size	Maximum *VAR* of smaller blocks inside the current CU (4×4 up to the current CU size, exclusive). *VAR_size* is used as VAR_32, VAR_16, VAR_8, and VAR_4 - the maxima variances of the 32×32, 16×16, 8×8, and 4×4 sub-blocks inside the current CU, respectively
Grad	Gradient of the 4 corners of the original CU (i.e., the maximum absolute difference among the 4 samples in the corners)
Grad_size	Maximum *Grad* of smaller blocks inside of the current CU. It can assume sizes similar to *Var_size*
MaxDiff	Maximum difference among samples of the current CU
RD-cost	RD-cost when encoding the current CU
QP	The current QP-depth value

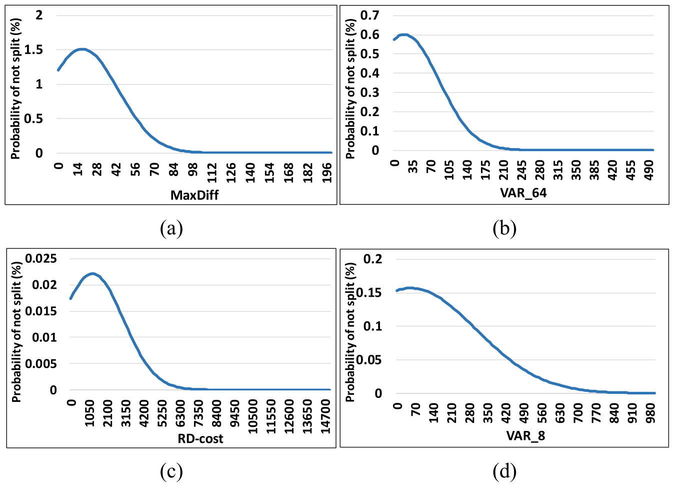


Fig. 7. Probability of not splitting 64 × 64 CU regarding the analyzed attributes for AI configuration.

were stored for each encoded block during the 3D-HTM execution. These attributes were selected intending to extract characteristics that have a high correlation with a splitting decision.

Fig. 7 exemplifies the probability density functions for four of the selected attributes, showing the correlation between these attributes and the CU splitting decision. Fig. 7(a), (b), (c) and (d) show *MaxDiff*, *VAR_64*, *RD-cost*, and *VAR_8* have lower values for those CUs not split into smaller ones. In these four cases, low values of attributes indicate a high probability of a decision that does not split the CU, while high values of these attributes indicate a high probability of splitting.

Among all evaluated attributes, some of them were selected, considering the correlation of each attribute with the CU split decision. The algorithm applied to build the static decision tree, which is based on the Information Gain (IG), defines the most relevant attributes for dealing with the split decision. IG refers to the difference between the entropy of all data set and the entropy of the subset of the evaluated attribute.

WEKA generated three intra-frames decision trees considering the IG of the analyzed attributes associated with the CU

TABLE III
ATTRIBUTES APPLIED TO I-FRAMES DECISION TREES USING *Kendo* AS TRAINING SEQUENCE

REP	Attribute	64×64		32×32		16×16	
		Used	IG	Used	IG	Used	IG
	QP	×	0.10	×	0.12	×	0.04
	RD-cost	×	0.31	×	0.20	×	0.29
	VAR_64	×	0.40	-		-	
	VAR_32	-		×	0.17	-	
	VAR_16	×	0.40	-		×	0.03
	VAR_8	-		×	0.17	×	0.03
	VAR_4	-		-		×	0.03
	MaxDiff	-		-		×	0.03
	Average	-		×	0.11	×	0.06
Yes	Accuracy	92.68%		83.59%		84.54%	
	Size	31		33		111	
No	Accuracy	93.65%		88.58%		88.11%	
	Size	4049		4491		11793	

split decision. Table III shows the complete list of attributes used to define the three intra-frame decision trees, the accuracy of the designed trees, the size regarding the number of nodes and the IG of each selected attribute using the *Kendo* video sequence. We also present the accuracy and the size of the tree, when REP is not used, showing a small increase in the accuracy with a high impact in the tree size.

Fig. 8 exemplifies the static decision tree generated for 64 × 64 CUs using *Kendo* as the training sequence, where the leaves "N" and "S" correspond to non-split and split decisions, respectively. The other eight trees generated focusing on I-frames (using one, two and four sequences for training process) were not shown in function of the available space in this article.

C. Unidirectional and Bi-Directional Frame Decision Trees

The decision trees employed on the P- and B-frames definition were built in a similar way to the ones used on the I-frames. The attributes analyzed for I-frames were also evaluated for P- and B-frames. However, new attributes were also assessed since there are many different features in P- and B-frames when compared to I-frames, such as inter-frame and inter-view dependencies. These new attributes and their description are listed in Table IV.

Fig. 9 exemplifies the probability density functions, regarding four of the selected attributes, showing the correlation between these attributes and the CU splitting decision. Fig. 9(a) shows the probability of the CU not be split according to the *MAD_4* (*MAD* of its 4 × 4 sub-blocks), Fig. 9(b) illustrates the probability of the CU not be split according to the *Neigh_depth*. Fig. 9(c) and (d) shows the probability of the CU be split according to the *Ratio* and *RelRatio*, respectively.

On the one hand, regarding *MAD_4* and *Neigh_depth* attributes, low values tend to keep the encoding CU size. On the other hand, low values of *Ratio* and *RelRatio* tends to split the encoding CU. Therefore, the knowledge of these attributes is crucial for achieving efficient decisions on the current CU splitting.

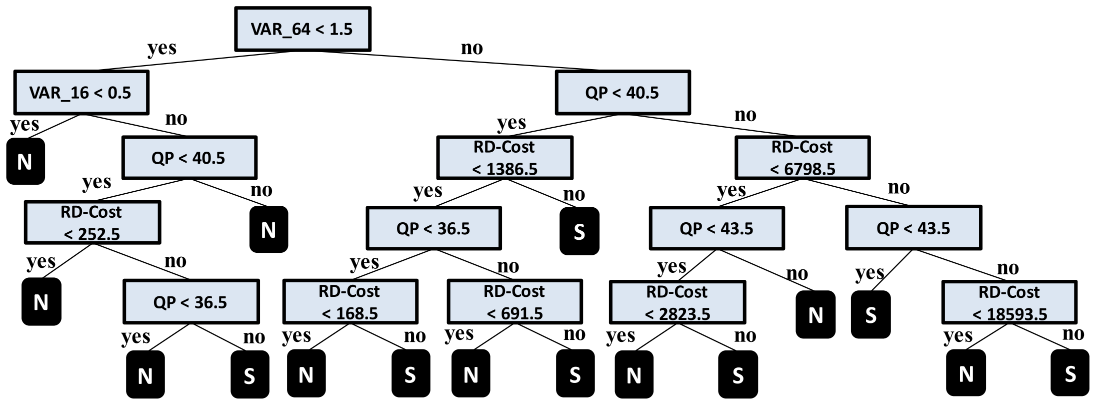


Fig. 8. Decision tree for splitting decision in 64 × 64 CUs for I-frames trees using *Kendo* as the training sequence.

TABLE IV
NEW ATTRIBUTES EVALUATED IN P- AND B-FRAMES

Attribute	Description
RD_MSM	RD-cost of *Merge*/*Skip* mode
RD_DIS	RD-cost obtained in DIS mode
Ratio	RD-cost obtained in intra-frame prediction divided by the RD_DIS
RatioInter	RD-cost of inter 2N×2N evaluation divided by RD_MSM
RelRatio	The normalized difference between the RD-cost of inter 2N×2N and RD_MSM
Neigh_depth	The average quadtree depth of the top, left, top-left, top-right, and co-located CTUs of both reference lists
SKIP_flag	Notify if the CU has been encoded using the *Skip* mode
DIS_flag	Notify if the CU has been encoded using the *DIS* mode
MAD_size	The maximum Mean Absolute Deviation of the smaller blocks inside of the current CU (4×4 up to current CU size, exclusive)

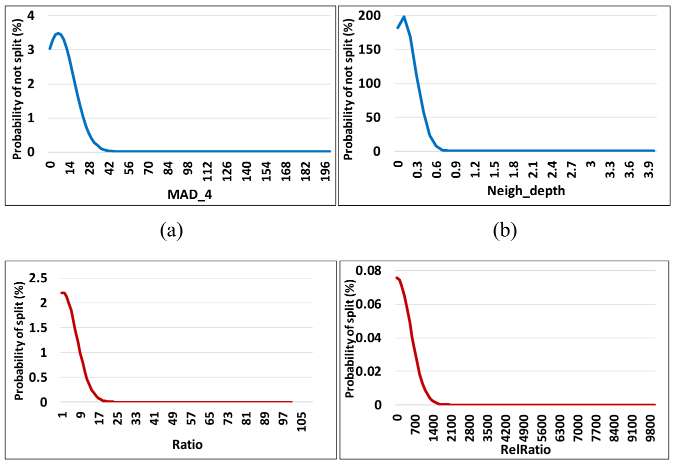


Fig. 9. Probability density function of the current 64 × 64 CU not be split (a) and (b) and be split (c) and (d) with RA configuration for four attributes.

Table V shows the attributes used in the three decision trees for P- and B-frames, the designed trees accuracy, the tree size and the IG of each attribute using *Kendo* as the training sequence. Again, the accuracy and size of not using REP are also available, justifying its usage.

TABLE V
ATTRIBUTES USED IN P- AND B-FRAMES DECISION TREES USING *Kendo* AS THE TRAINING SEQUENCE

REP	Attribute	64×64		32×32		16×16	
		Used	IG	Used	IG	Used	IG
	QP	-		×	0.03	×	0.02
	RD-cost	×	0.19	×	0.22	×	0.22
	RD_MSM	×	0.20	×	0.18	×	0.15
	RD_DIS	-		×	0.21	-	
	Ratio	×	0.50	×	0.23	×	0.23
	RatioInter	-		×	0.19	-	
	RelRatio	×	0.48	×	0.19	×	0.14
	Neigh_depth	×	0.49	-		-	
	SKIP_flag	×	0.26	×	0.07	×	0.01
	DIS_flag	-		×	0.06	-	
	VAR_64	×	0.27	-		-	
	VAR_32	-		×	0.21	-	
	VAR_16	×	0.27	-		-	
	MAD_4	×	0.27	-		-	
	MaxDiff	×	0.28	-		-	
Yes	Accuracy	91.65%		83.08%		82.13%	
Yes	Size	33		59		37	
No	Accuracy	92.85%		85.11%		83.68%	
No	Size	1523		2831		2291	

Fig. 10 exemplifies the decision tree built for 64 × 64 CUs using *Kendo* as the training sequence. The remaining decision trees (using one, two and four sequences for training process) were not shown in this article in function of the available space.

VI. EXPERIMENTAL RESULTS

The decision trees were implemented inside the 3D-HTM version 16.0 [17] and evaluated following the CTC for 3D experiments [47], which have the same configurations presented in Table I. However, in this case, the sequences used in the training process were not considered in the evaluating process.

Firstly, the decision trees were evaluated using the AI configuration. Next, the performance of the proposed solution was evaluated using RA configuration. The results employ Bjontegaard Delta-rate (BD-rate) [59] metric considering

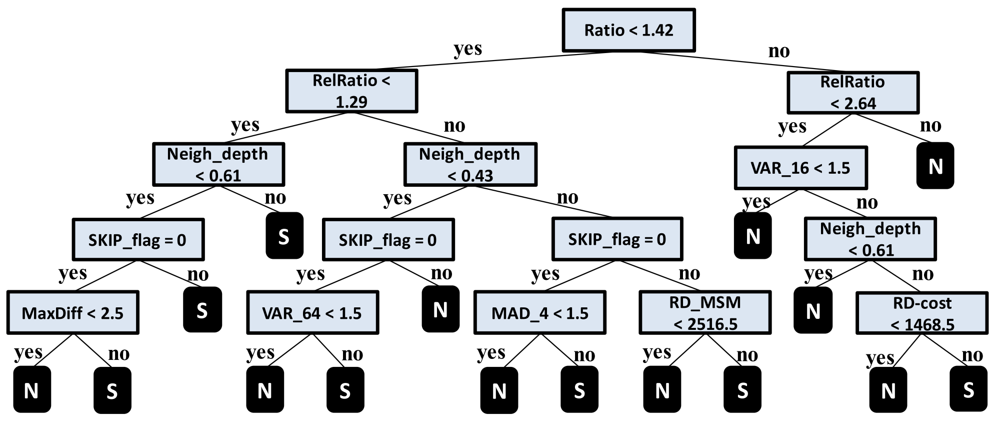


Fig. 10. Decision tree for splitting decision in 64 × 64 CUs for P- and B-frames using *Kendo* as the training sequence.

TABLE VI
EXPERIMENTAL RESULTS WITH THE AI CONFIGURATION

Video	DT-1		DT-2		DT-4	
	BDRI (%)	CER (%)	BDRI (%)	CER (%)	BDRI (%)	CER (%)
Balloons	0.14	45.7	0.09	46.0	0.05	40.7
Newspaper_CC	0.11	37.3	0.19	40.7	0.14	37.6
GT_Fly	0.06	58.8	0.04	59.7	0.06	58.4
Poznan_Hall2	0.62	63.4	0.53	62.5	0.66	60.3
Undo_Dancer	0.14	56.5	0.25	55.9	0.27	54.6
Shark	0.04	51.9	0.02	46.7	0.19	50.9
Average	0.19	52.3	0.19	51.9	0.23	50.4

TABLE VII
COMPARISON WITH RELATED WORK IN AI CONFIGURATION

Work		BDRI (%)	CER (%)
Shen [18]		0.20	-
Zhang [19]		-0.15	30.0
Chen [20]		0.00	22.8
Peng [22]		0.80	37.7
Zhang [23]		0.44	41.0
This work	DT-1	0.19	52.3
	DT-2	0.19	51.9
	DT-4	0.23	50.4

the synthesized views quality and the computational effort reduction of the entire encoder (texture and depth maps). Section VI-A shows the results and comparisons for AI configuration, while Section VI-B compares the results obtained with the RA configuration.

A. All Intra (AI) Configuration Results

Table VI shows the results reached by the decision trees specialized in intra-frames. This experiment was done considering the AI configuration. The decision trees created using *Kendo* (DT-1) sequence, the decision trees created with *Kendo* and *Poznan_Street* (DT-2), and the decision trees created with *Kendo*, *Poznan_Street*, *Champagne_tower*, and *Pantomime* (DT-4) sequences were evaluated. The results are presented using two main metrics: BD-Rate Increase (BDRI) and Computational Effort Reduction (CER). These results consider the complete encoder (texture plus depth encoding).

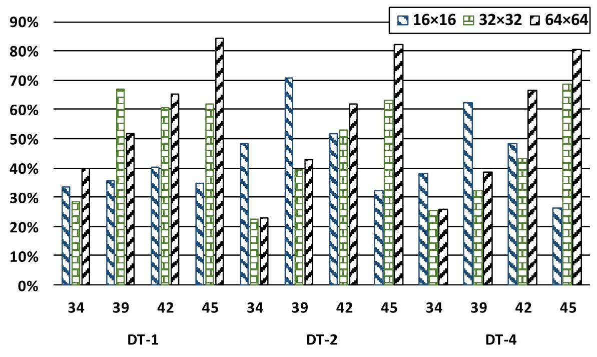
Table VI shows an average computational effort reduction of 52.3%, 51.9% and 50.4% using DT-1, DT-2 and DT-4, respectively. As a drawback, the proposed solution presented a BD-Rate increase of 0.19% (DT-1 and DT-2) and 0.23% (DT-4). Additionally, when considering only depth map coding, the proposed solution reduces 59.0% (DT-1), 58.5% (DT-2) and 55.8% (DT-4) the encoding effort, on average.

Fig. 11(a) illustrates the percentage of CUs that were not split according to the CU size and QP-depth value under AI configuration. As higher is the QP-depth, as higher is the probability of the CUs to be encoded with larger sizes; consequently, as higher is the percentage of not splitting decisions. It happens because higher QP-depth values tend to generate more homogeneous regions (with fewer details), and homogeneous regions are more efficiently encoded using larger CU sizes. For instance, the percentage of CUs that were not split with 64×64 CUs and QP-depth = 45 is 84.52% (DT-1), 82.25% (DT-2) and 80.51% (DT-4), on average.

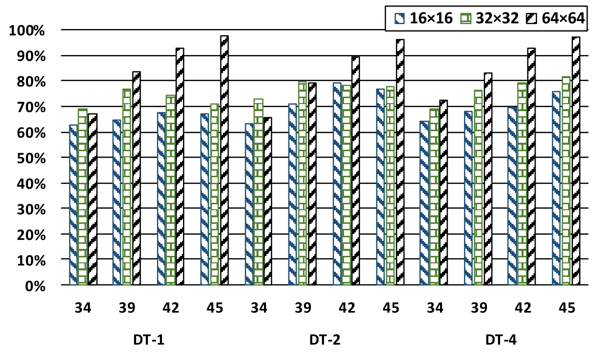
The reached results for AI configuration, including those presented in Table VI and Fig. 11(a), one can conclude our methodology is robust independently of the number of used training sequences since similar results were reached when using one, two or four training sequences.

Finally, we emphasize the proposed solution can halve the computational effort of the entire 3D-HEVC encoder (texture plus depth) at the cost of less than 0.23% in BD-rate, on average.

Table VII compares the proposed solution with the works [18]–[20], [22], and [23], which also focus in AI configuration. In [18], CER is only presented for depth maps, where 61% is obtained with 0.2% of BDRI. BDRI of our solution is in the range [0.19%, 0.23%] with a CER of depth maps in the range [55.8%, 59.0%]; therefore, these results are quite similar. The methods proposed by [19], [20], [22], and [23] have a CER ranging from 22.8% to 41% with BDRI ranging from −0.15% to 0.8%. Our solution obtained a CER and BDRI in the ranges [50.4%, 52.3%] and [0.19%, 0.23%], respectively, on average. Thus, our solution presents very competitive results when compared to these related works in both metrics.



(a)



(b)

Fig. 11. Percentage of non-splitting CUs according to the block size and QP-depth for (a) AI and (b) RA configurations.

TABLE VIII

EXPERIMENTAL RESULTS WITH RA CONFIGURATION

Video	DT-1		DT-2		DT-4	
	BDRI (%)	CER (%)	BDRI (%)	CER (%)	BDRI (%)	CER (%)
Balloons	0.06	47.7	0.11	53.3	0.09	54.3
Newspaper_CC	0.05	48.1	0.05	55.3	0.03	57.0
GT_Fly	0.07	52.2	-0.02	57.4	0.03	57.7
Poznan_Hall2	0.39	59.4	0.21	64.6	0.28	64.7
Undo_Dancer	0.55	53.3	0.12	57.7	0.34	58.4
Shark	0.01	51.4	-0.05	57.2	0.00	57.6
Average	0.19	52.0	0.07	57.6	0.13	58.3

TABLE IX

COMPARISON WITH RELATED WORK IN RA CONFIGURATION

Work		BDRI (%)	CER (%)
Shen [18]		0.10	-
Zhang [19]		0.08	4.9
Zhang [21]		0.78	43.2
Mora [24]		1.02	52.0
Lei [25]		2.06	41.5
This work	DT-1	0.19	52.0
	DT-2	0.07	57.6
	DT-4	0.13	58.3

B. Random-Access (RA) Configuration Results

The entire solution of fast 3D-HEVC depth maps was evaluated using the RA configuration, where all proposed decision trees are used together. Table VIII summarizes the experimental results with the RA configuration showing DT-1 reached a CER of 52.0% and a BDRI of 0.19%, DT-2 obtained a CER of 57.6% and a BDRI of 0.07%, and DT-4 achieved a CER of 58.3% and a BDRI of 0.13%. Moreover, regarding only depth map coding our solution reached a CER of 68.0% (DT-1), 75.8% (DT-2) and 76.7% (DT-4), on average.

Fig. 11(b) depicts the percentage of CUs were not split according to the CU size and QP-depth value when evaluated in RA configuration. The tendencies are similar to those presented in Fig. 11(a) but less pronounced in this case. Again, as higher is the QP-depth value, as higher is the use of larger CUs; consequently, as higher is the CUs with not split decisions. The highest not splitting percentage reaches 97.7% (DT-1), 96.2% (DT-2) and 97.5% (DT-4) with QP-depth = 45 when processing 64×64 CUs.

Table IX compares the solution proposed in this article with the works [18], [19], [21], [24], and [25], which also focus on RA configuration. Again, in [18], CER is presented only for depth map coding, where 40% is obtained with a BDRI of 0.1%. Our decision trees reached average results of BDRI and CER (when considering only the depth maps) in the ranges [0.07%, 0.19%] and [68.0%, 76.7%], respectively. When compared to [19], our proposed solution achieved similar results of BDRI, but with a CER more than ten times better. The solutions presented in [21] and [25] achieve good results of CER; however, with significant losses on BDRI. Our decision trees reached a little better CER results than [21] and [25], but with a much lower impact in BDRI. Finally, the state-of-the-art method presented in [24] (adopted in 3D-HTM) reached 52.0% of CER with BDRI of 1.02%, whereas the proposed decision trees present better results in both axes.

The experiments using RA configuration showed the decision trees reached very competitive results where only one related work [24] reached the same CER, but with 6.8 times more impact on BD-rate, when compared to DT-1. On the other hand, only one work [19] reached similar BD-rate impacts, but with a CER more than 10 times smaller.

The experiments demonstrated that using a different number of training sequences tends to achieve similar results of non-splitting CUs with small variations in BD-rate and computational effort. Besides, one can notice a higher number of training sequences does not ensure better results, as demonstrated in Table VI and Table VIII. One can conclude the proposed solution is robust and presented excellent results even when used one, two or four training sequences.

VII. CONCLUSIONS

This article presented a fast 3D-HEVC depth map encoding solution, which uses static decision trees to define the partitioning of CUs instead of the complete and complex RDO process. After encoding a given CU size, a decision tree decides if the CU must be split or not into smaller sizes. The decision trees were built with data mining and machine learning to extract correlations among the encoder context attributes.

Two types of trees were defined using J48 algorithm available on the WEKA software, being one for I-frames and other for P- and B-frames since the encoder behavior is significantly different for these types of frames, requiring specialized trees. Besides, one decision tree was created for each one of the three levels of CU sizes where the partition is allowed (16×16, 32×32 and 64×64), indicating if the CU must or not be split into smaller CUs. Finally, three sets of decision trees were evaluated, one set built using only one training sequence, another set built using two training sequences, and a set built using four training sequences, trying to evaluate the robustness of the proposed approach. Then, a total of 18 decision trees were built and evaluated in this article.

The evaluation of the developed trees was done using a set of six sequences that were not used in the training process. Through experimental analysis, the proposed solution reached an average Computational Effort Reduction (CER) higher than 50% for AI and RA configurations. In both cases, the BD-rate has increased by less than 0.25% in the synthesized views, on average. The best results were reached by the decision trees built using two sequences in the training process and when these trees were evaluated through the RA configuration, where the trees for I-frames and the trees for P- and B-frames were used together. In this case, our solution can reduce in almost 58% the complete 3D-HEVC encoder computational effort (texture plus depth) with a BD-rate drop of only 0.07%, surpassing all related works in both axes.

Considering the presented results, we conclude the proposed solution can be efficiently used to encode depth maps with better results than all published solutions, including those solutions implemented in the 3D-HTM.

References

[1] O. Stankiewicz, M. Domański, A. Dziembowski, A. Grzelka, D. Mieloch, and J. Samelak, "A free-viewpoint television system for horizontal virtual navigation," *IEEE Trans. Multimedia*, vol. 20, no. 8, pp. 2182–2195, Aug. 2018.

[2] 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.

[3] K. Müller *et al.*, "3D high-efficiency video coding for multi-view video and depth data," *IEEE Trans. Image Process.*, vol. 22, no. 9, pp. 3366–3378, May 2013.

[4] G. Tech, Y. Chen, K. Müller, J.-R. Ohm, A. Vetro, and Y.-K. Wang, "Overview of the multiview and 3D extensions of high efficiency video coding," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 26, no. 1, pp. 35–49, Jan. 2016.

[5] H. Zhang and Z. Ma, "Fast intra mode decision for High Efficiency Video Coding (HEVC)," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 24, no. 4, pp. 660–668, Apr. 2014.

[6] B. Min and R. C. C. Cheung, "A fast CU size decision algorithm for the HEVC intra encoder," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 25, no. 5, pp. 892–896, May 2015.

[7] S. Ahn, B. Lee, and M. Kim, "A novel fast CU encoding scheme based on spatiotemporal encoding parameters for HEVC inter coding," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 25, no. 3, pp. 422–435, Mar. 2015.

[8] L. Shen, Z. Zhan, and Z. Liu, "Adaptive inter-mode decision for HEVC jointly utilizing inter-level and spatiotemporal correlations," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 24, no. 10, pp. 1709–1722, Oct. 2014.

[9] P. Kauff *et al.*, "Depth map creation and image-based rendering for advanced 3DTV services providing interoperability and scalability," *Signal Process., Image Commun.*, vol. 22, no. 2, pp. 217–234, Feb. 2007.

[10] M. Saldanha, G. Sanchez, B. Zatt, M. Porto, and L. Agostini, "Complexity reduction for the 3D-HEVC depth maps coding," in *Proc. IEEE Int. Symp. Circuits Syst. (ISCAS)*, May 2015, pp. 621–624.

[11] S. Ma, S. Wang, and W. Gao, "Low complexity adaptive view synthesis optimization in HEVC based 3D video coding," *IEEE Trans. Multimedia*, vol. 16, no. 1, pp. 266–271, Jan. 2014.

[12] M.-K. Kang and Y.-S. Ho, "Depth video coding using adaptive geometry based intra prediction for 3-D video systems," *IEEE Trans. Multimedia*, vol. 14, no. 1, pp. 121–128, Feb. 2012.

[13] J. Lei, C. Zhang, Y. Fang, Z. Gu, N. Ling, and C. Hou, "Depth sensation enhancement for multiple virtual view rendering," *IEEE Trans. Multimedia*, vol. 17, no. 4, pp. 457–469, Apr. 2015.

[14] P. Merkle, K. Müller, D. Marpe, and T. Wiegand, "Depth intra coding for 3D video based on geometric primitives," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 26, no. 3, pp. 570–582, Mar. 2015.

[15] J. Lee, M. W. Park, and C. Kim, *3D-CE1: Depth Intra Skip (DIS) Mode*, document JCT3V-K0033, Geneva, Switzerland, Feb. 2015.

[16] H. Liu and Y. Chen, "Generic segment-wise DC for 3D-HEVC depth intra coding," in *Proc. IEEE Int. Conf. Image Process. (ICIP)*, Oct. 2014, pp. 3219–3222.

[17] *3D-HEVC Test Model*. Accessed: Jul. 2018. [Online]. Available: https://hevc.hhi.fraunhofer.de/svn/ svn_3DVCSoftware/tags/HTM-16.0/

[18] L. Shen, K. Li, G. Feng, P. An, and Z. Liu, "Efficient intra mode selection for depth-map coding utilizing spatiotemporal, inter-component and inter-view correlations in 3D-HEVC," *IEEE Trans. Image Process.*, vol. 27, no. 9, pp. 4195–4206, Sep. 2018.

[19] Q. Zhang, Y. Yang, H. Chang, W. Zhang, and Y. Gan, "Fast intra mode decision for depth coding in 3D-HEVC," *Multidimensional Syst. Signal Process.*, vol. 28, no. 4, pp. 1203–1226, Feb. 2016.

[20] J. Chen, B. Wang, H. Zeng, C. Cai, and K.-K. Ma, "Sum-of-gradient based fast intra coding in 3D-HEVC for depth map sequence (SOG-FDIC)," *J. Vis. Commun. Image Represent.*, vol. 48, pp. 329–339, Oct. 2017.

[21] Q. Zhang, N. Zhang, T. Wei, K. Huang, X. Qian, and Y. Gan, "Fast depth map mode decision based on depth–texture correlation and edge classification for 3D-HEVC," *J. Vis. Commun. Image Represent.*, vol. 45, pp. 170–180, May 2017.

[22] K.-K. Peng, J.-C. Chiang, and W.-N. Lie, "Low complexity depth intra coding combining fast intra mode and fast CU size decision in 3D-HEVC," in *Proc. IEEE Int. Conf. Image Process. (ICIP)*, Sep. 2016, pp. 1126–1130.

[23] H.-B. Zhang, Y.-L. Chan, C.-H. Fu, S.-H. Tsang, and W.-C. Siu, "Quadtree decision for depth intra coding in 3D-HEVC by good feature," in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process. (ICASSP)*, Mar. 2016, pp. 1481–1485.

[24] E. G. Mora, J. Jung, M. Cagnazzo, and B. Pesquet-Popescu, "Initialization, limitation, and predictive coding of the depth and texture quadtree in 3D-HEVC," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 24, no. 9, pp. 1554–1565, Sep. 2014.

[25] J. Lei, J. Duan, F. Wu, N. Ling, and C. Hou, "Fast mode decision based on grayscale similarity and inter-view correlation for depth map coding in 3D-HEVC," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 28, no. 3, pp. 706–718, Mar. 2016.

[26] H.-B. Zhang, C.-H. Fu, Y.-L. Chan, S.-H. Tsang, and W.-C. Siu, "Probability-based depth intra-mode skipping strategy and novel VSO metric for DMM decision in 3D-HEVC," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 28, no. 2, pp. 513–527, Feb. 2018.

[27] C.-H. Fu, H.-B. Zhang, W.-M. Su, S.-H. Tsang, and Y.-L. Chan, "Fast wedgelet pattern decision for DMM in 3D-HEVC," in *Proc. IEEE Int. Conf. Digit. Signal Process. (DSP)*, Jul. 2015, pp. 477–481.

[28] S. Jaballah, M.-C. Larabi, and J. B. Tahar, "Heuristic inspired search method for fast wedgelet pattern decision in 3D-HEVC," in *Proc. 6th Eur. Workshop Vis. Inf. Process. (EUVIP)*, 2016, pp. 1–6.

[29] H.-B. Zhang, S.-H. Tsang, Y.-L. Chan, C.-H. Fu, and W.-M. Su, "Early determination of intra mode and segment-wise DC coding for depth map based on hierarchical coding structure in 3D-HEVC," in *Proc. Asia-Pacific Signal Inf. Process. Assoc. Annu. Summit Conf. (APSIPA)*, 2015, pp. 374–378.

[30] S. Ma, Y. Wang, C. Zhu, Y. Lin, and J. Zheng, "Reducing Wedgelet lookup table size with down-sampling for depth map coding in 3D-HEVC," in *Proc. IEEE Int. Workshop Multimedia Signal Process. (MMSP)*, Oct. 2015, pp. 1–5.

[31] M. Kim, N. Ling, L. Song, "Fast single depth intra mode decision for depth map coding in 3D-HEVC," in *Proc. IEEE Int. Conf. Multimedia Expo Workshops (ICMEW)*, Jun./Jul. 2015, pp. 1–6.

[32] G. Sanchez, M. Saldanha, G. Balota, B. Zatt, M. Porto, and L. Agostini, "Complexity reduction for 3D-HEVC depth maps intra-frame prediction using simplified edge detector algorithm," in *Proc. IEEE Int. Conf. Image Process. (ICIP)*, Oct. 2014, pp. 3209–3213.

[33] Z. Liu, X. Yu, Y. Gao, S. Chen, X. Ji, and D. Wang, "CU partition mode decision for HEVC hardwired intra encoder using convolution neural network," *IEEE Trans. Image Process.*, vol. 25, no. 11, pp. 5088–5103, Nov. 2016.
[34] G. Correa, P. A. Assuncao, L. V. Agostini, and L. A. da Silva Cruz, "Fast HEVC encoding decisions using data mining," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 25, no. 4, pp. 660–673, Apr. 2015.
[35] D. Ruiz-Coll, V. Adzic, G. Fernández-Escribano, H. Kalva, J. L. Martínez, P. Cuenca, "Fast partitioning algorithm for HEVC Intra frame coding using machine learning," in *Proc. IEEE Int. Conf. Image Process. (ICIP)*, Oct. 2014, pp. 4112–4116.
[36] F. Duanmu, Z. Ma, and Y. Wang, "Fast mode and partition decision using machine learning for intra-frame coding in HEVC screen content coding extension," *IEEE J. Emerg. Sel. Topics Circuits Syst.*, vol. 6, no. 4, pp. 517–531, Dec. 2016.
[37] E. Peixoto, B. Macchiavello, R. L. de Queiroz, and E. M. Hung, "Fast H.264/AVC to HEVC transcoding based on machine learning," in *Proc. Int. Telecommun. Symp. (ITS)*, 2014, pp. 1–4.
[38] M. Saldanha, G. Sanchez, C. Marcon, and L. Agostini, "Fast 3D-HEVC depth maps intra-frame prediction using data mining," in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process. (ICASSP)*, Apr. 2018, pp. 1738–1742.
[39] Y. Chen, X. Zhao, L. Zhang, and J.-W. Kang, "Multiview and 3D video compression using neighboring block based disparity vectors," *IEEE Trans. Multimedia*, vol. 18, no. 4, pp. 576–589, Apr. 2016.
[40] G. J. Sullivan, J. M. Boyce, Y. Chen, J.-R. Ohm, C. A. Segall, and A. Vetro, "Standardized extensions of High Efficiency Video Coding (HEVC)," *IEEE J. Sel. Topics Signal Process.*, vol. 7, no. 6, pp. 1001–1016, Dec. 2013.
[41] G. Sanchez, J. Silveira, L. Agostini, and C. Marcon, "Performance analysis of depth intra coding in 3D-HEVC," *IEEE Trans. Circuits Syst. Video Technol.*, to be published.
[42] X. Ji, D. Zhao, W. Gao, Q. Huang, S. Ma, and Y. Lu, "New bi-prediction techniques for B pictures coding," in *Proc. IEEE Int. Conf. Multimedia Expo (ICME)*, Jun. 2004, pp. 101–104.
[43] P. Helle *et al.*, "Block merging for quadtree-based partitioning in HEVC," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 22, no. 12, pp. 1720–1731, Dec. 2012.
[44] F. Jäger, "Depth-based block partitioning for 3D video coding," *Int. Picture Coding Symp. (PCS)*, 2013, pp. 410–413.
[45] *Advanced Video Coding for Generic Audiovisual Services*, Standard ITU-T Rec. H.264 and ISO/IEC 14496-10:2005, EMPEG-4 AVC, 2005.
[46] C. Apté and S. Weiss, "Data mining with decision trees and decision rules," *Future Gener. Comput. Syst.*, vol. 13, no. 2, pp. 197–210, Nov. 1997.
[47] K. Müller and A. Vetro, *Common Test Conditions of 3DV Core Experiments*, document ISO/IEC JTC1/SC29/WG11 MPEG2011/N12745, Jan. 2014.
[48] *Tanimoto Lab*. Accessed: Jul. 2018. [Online]. Available: http://www.tanimoto.nuee.nagoya-u.ac.jp/
[49] Y.-S. Ho, E.-K. Lee, and C. Lee, *Multiview Video Test Sequence and Camera Parameters*, document ISO/IEC JTC1/SC29/WG11 MPEG2008/M15419, Archamps, France, Apr. 2008.
[50] J. Zhang, R. Li, H. Li, D. Rusanovskyy, and M. M. Hannuksela, *Ghost Town Fly 3DV Sequence for Purposes of 3DV Standardization*, document ISO/IEC JTC1/SC29/WG11 MPEG, Doc. M20027, Geneva, Switzerland, Mar. 2011.
[51] M. Domañski *et al.*, *Poznan multiview Video Test Sequences and Camera Parameters*, document ISO/IEC JTC1/SC29/WG11 MPEG 2009/M17050, Xian, China, Oct. 2009.
[52] D. Rusanovskyy, P. Aflaki, and M. M. Hannuksela, *Undo Dancer 3DV Sequence for Purposes of 3DV Standardization*, document ISO/IEC JTC1/SC29/WG11 MPEG, Doc. M20028, Geneva, Switzerland, Mar. 2011.
[53] *National Institute of Information and Communication Technology (NICT)*. Accessed: Jul. 2018. [Online]. Available: ftp://ftp.merl.com/pub/tian/NICT-3D/Shark/
[54] G. Chi, X. Jin, and Q. Dai, "A Quad-Tree and Statistics based fast CU Depth decision algorithm for 3D-HEVC," in *Proc. IEEE Int. Conf. Multimedia Expo Workshops (ICMEW)*, Jul. 2014, pp. 1–5.
[55] C. Harris and J. Stephens, "A combined corner and edge detector," in *Proc. Alvey Vis. Conf.*, 1988, pp. 147–151.
[56] M. Hall, E. Frank, G. Holmes, B. Pfahringer, P. Reutemann, and I. H. Witten, "The WEKA data mining software: An update," *ACM SIGKDD Explor. Newslett.*, vol. 11, no. 1, pp. 10–18, 2009.
[57] J. R. Quinlan, "C4.5: Programs for machine learning," Morgan Kaufmann Publishers, San Francisco, CA, USA, 1993.
[58] C. A. Brunk and M. J. Pazzani, "An investigation of noise-tolerant relational concept learning algorithms," in *Proc. 8th Int. Conf.*, 1991, pp. 389–393.
[59] G. Bjontegaard, "Calculation of average PSNR differences between RD-Curves," in *Proc. VCEG Meeting*, 2001, pp. 1–5.

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

Gustavo Sanchez received the Electrical Engineering degree from the Sul-Rio-Grandense Federal Institute of Education, Science and Technology, in 2013, the B.S. degree in computer science from the Federal University of Pelotas, in 2012, and the M.Sc. degree in computer science from the Federal University of Pelotas, in 2014. He is currently pursuing the Ph.D. degree in computer science with the Pontifical Catholic University of Rio Grande do Sul. He has been a Professor with IF Farroupilha, Brazil, since 2014. He has over eight 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 and 3D video coding.

César Marcon 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 School of Computer Science, Pontifical Catholic University of Rio Grande do Sul (PUCRS), Brazil, since 1995. He is an Advisor of M.Sc. and Ph.D. graduate students at the Graduate Program in Computer Science of PUCRS. He has more than 100 papers published in prestigious journals and conference proceedings. Since 2005, he has coordinated nine research projects in the areas of telecom, healthcare, and telemedicine. His research interests include embedded systems in the telecom domain, MPSoC architectures, partitioning and mapping application tasks, and fault-tolerance and real-time operating systems. He is a member of the IEEE and the Brazilian Computer Society.

Luciano Agostini (M'06–SM'11) 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 has been a Professor with the Federal University of Pelotas (UFPel), Brazil, since 2002, where he leads the Video Technology Research Group. He was the Executive Vice President for Research and Graduate Studies of UFPel, from 2013 to 2017. He advises master's and doctorate students at the UFPel Graduate Program in Computer. He has more than 200 published papers 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 the 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 the SBC and SBMicro Brazilian societies.