Low Complexity Versatile Video Coding (VVC) for Low Bitrate Applications

Mourad Aklouf, Marc Leny, Frédéric Dufaux, Michel Kieffer

LOW COMPLEXITY VERSATILE VIDEO CODING (VVC) FOR LOW BITRATE APPLICATIONS

Mourad AKLOUF, Marc LENY

{maklouf, mleny}@ektacom.com,
EKTACOM,
91940 Les Ulis, France.

Frederic DUFAUX, Michel KIEFFER

{frederic.dufaux, michel.kieffer}@l2s.centralesupelec.fr,
L2S - CentraleSupelec - CNRS - Univ Paris Sud,
91192 Gif-sur-Yvette, France.

ABSTRACT

The new Versatile Video Coding (VVC) standard is currently under development, targeting more efficient compression, especially for 4K and 8K contents. Nevertheless, some application scenarios still require low resolution and low bitrate encoding, *e.g.*, live video streaming over unstable bandwidth-limited wireless networks. In addition, embedded applications also require low complexity and low power consumption solutions. As VVC is not designed for these conditions, it may not necessarily achieve the optimal trade-off between complexity and compression efficiency. This paper proposes an optimization framework to determine the efficiency at low-resolution and low-bitrate of some of the new coding tools introduced in VVC. We show that in such coding conditions, significant reduction in terms of encoding complexity (up to 56% at 384×216 resolution) may be obtained by disabling some of the VVC coding tools, with a negligible impact in terms of compression efficiency (less than 1.88% increase in BD_{rate}).

***Index Terms*—** Versatile Video Coding, video compression, optimization, low complexity, low resolution, low bitrate

1. INTRODUCTION

Video traffic over Internet is continuously increasing. Video is foreseen to represent 82% of all consumer Internet traffic by 2021, 26% of which will be Internet video-to-TV traffic. Moreover, Content Delivery Network (CDN) traffic will carry 71% of all Internet traffic by 2021 [1].

This congestion is mainly due to users expecting better quality of experience, including High Definition (HD) and Ultra-High Definition (UHD) video content. UHD, including 3840×2160 (4K) and 7680×4320 (8K) formats, is expected to become mainstream. For instance, 4K TVs are already available since 2014, and 8K consumer TVs are expected in 2020. Even portable devices are expected to handle UHD in the next few years, due to increasing hardware power and efficiency of IP networks. In this context, the need for new efficient video compression tools is clear, on one hand to decrease the load on transmission channels and storage servers, and on the other hand to provide high-quality content to end-users.

The High-Efficiency Video Coding (HEVC) standard [2] is the latest standard developed by the Joint Collaborative Team on Video Coding (JCT-VC) of VCEG and MPEG. It is effectively deployed worldwide and compares favorably with competing solutions [3]. Nevertheless, more efficient video compression techniques are still desired. With this goal in mind, the new Versatile Video Coding (VVC) standard is currently under development by the Joint Video Exploration Team (JVET) of VCEG and MPEG and is expected to be finalized by 2020 [4].

While the above standards are mainly focused on HD and UHD content, efficient video compression solutions are also requested for streaming Standard Definition (SD) video (or even lower resolutions) over unstable bandwidth-limited networks. Indeed, whereas HD video is now the standard on the Internet, low-resolution contents are still used, especially 480p and 360p. In this paper, we specifically consider use cases such as the acquisition and live streaming of low-resolution (less than HD) sport events (*e.g.*, car races and sailing races) over relatively unstable wireless networks including LTE and satellite transmissions. For such use cases, transmission bandwidth when using common wireless networks is typically in the range of 50 to 700 kbps. Moreover, low computational complexity and low power consumption solutions are highly desired for embedded systems.

As current VVC developments are not designed for such use cases, some of the design choices for HD and UHD may not be optimal for lower resolutions and low-bitrates. More precisely, a variety of encoding tools available at the VVC encoder may entail a significant burden in terms of computational complexity. Moreover, some of these tools may not be suited in the case of low resolution and low bitrate scenarios. In particular, the cost of syntax elements to signal the activation of these tools may represent a non-negligible overhead at low bitrates.

The contribution of this paper is two fold. We first propose an optimization framework with the objective to tune VVC for low-resolution and low-bitrate scenarios. More specifically, we investigate the usefulness of some of the new coding tools in VVC. Second, we experimentally show that significant complexity reduction can be achieved (up to 56.06% reduction for *Johnny* sequence at 384×216 resolution) by disabling some of these tools, while preserving coding efficiency (less than 1.88% increase in BD_{rate}). To the best of our knowledge, this is the first study to investigate VVC at low resolutions and low-bitrates, and to report such complexity reduction results.

The remaining of this paper is structured as follow. Some related works are first reviewed in Section 2. Section 3 provides a brief overview of the technical features of the VVC standard and the coding tools in the VVC Test Model (VTM). The problem formulation and proposed optimization framework are introduced in Section 4. Experimental results are reported in Section 5. Finally, we draw some conclusions in Section 6.

2. RELATED WORKS

The problem of encoder optimization has already been largely addressed in the literature. In particular, many proposals have considered the simplification of the Rate Distortion Optimization (RDO) process of the HEVC encoder. Targeting the Coding Unit (CU) size and depth decision, an algorithm to quickly determine the CU size has been proposed in [5] for both Intra and Inter modes. To reduce the number of Intra mode candidates, [6] uses the coding information of the neighboring blocks. Conversely, [7] introduces a decision algorithm, which compute the dominant edge of the Prediction Unit (PU) using the local samples. Targeting the Motion Vectors (MVs) search in Inter prediction, [8] presents an heuristic to determine the PU partition by checking all CUs among the neighboring NxN partition.

All the previous approaches make algorithmic modifications to the non-normative encoding process. In this paper, our aim is to identify coding tools which can be ignored in low-bitrate use cases, with a greatly reduced complexity and a preserved coding efficiency. This could lead to the definition of application-oriented profiles, where some tools are automatically disabled in the high-level syntax.

3. VERSATILE VIDEO CODING (VVC)

VVC [9] is a new video coding standard designed by JVET, which goal is to provide significant improvements in compression performance over the existing HEVC standard, with a target of 50% bitrate saving. VVC is expected to deliver UHD services (4K and 8K) at approximately the same bit rates used today to carry HDTV. The requirements for VVC include capabilities of encoding 4K and 8K sequences of up to 120 fps [10]. In what follows, some of the coding tools in VTM5.0 are overviewed [11].

3.1. Partitioning

Each frame is divided into a sequence of coding tree units (CTUs) just as in the HEVC standard, although the maximum size of the Luma CTU is up to 128×128. For each CTU, a Quad-Tree with nested Multi-Type Tree (MTT) using Binary and Ternary splitting structures is used (QTBT-TT). The CTU is first partitioned recursively using a quad-tree structure into square shapes. Then, the quad-tree leaf nodes can be further partitioned horizontally or vertically by a binary or ternary splitting structure. The final nodes are called Coding Units (CUs). They have either a square or rectangular shape and are used directly for prediction and residual coding. Lastly, I-slices can have separate block tree structures for Luma and Chroma (DualTree).

3.2. Intra-Picture Prediction

VTM5.0 supports 65 angular intra-prediction modes, in addition to the planar and DC modes. Some conventional angular modes are replaced with wide-angle intra-prediction modes for the non-square blocks. A Multiple Reference Line (MRL) intra prediction is also proposed to use two additional lines (reference line 1 and reference line 3) in angular prediction. VTM5.0 also extends the Most Probable Modes (MPM) list to 6 candidates.

VTM5.0 introduces three new ways of Intra predicting a block: (1) the Cross-Component Linear Model (CCLM) prediction mode, in which the Chroma samples are predicted based on the reconstructed Luma samples of the same CU, using a linear model; (2) The Intra Sub-Partitions (ISP) where the Luma CB is vertically or horizontally divided into 2 or 4 sub-partitions. All sub-partitions share the same intra mode, however the processing is performed gradually sub-partition by sub-partition downwards (horizontal split) or rightwards (vertical split), so each one uses the previous reconstructed samples to generate the prediction of the current sub-partition; (3) The Matrix-based Intra Prediction (MIP) takes one line of reconstructed neighboring samples, from the left and above blocks as input vectors, and after some pre-processing, performs linear interpolation in the vertical and horizontal directions.

3.3. Inter-Picture Prediction

Motion prediction is performed at a sub-CU level to improve the precision. VTM5.0 supports currently a Sub-Pu Temporal Motion Vector Prediction (SbTMVP). Furthermore, an AFFine motion compensation prediction (AFF) can be applied to cope with irregular motions like zoom in/out and rotation, where a sub-block is described by two or three motion vectors.

The bi-prediction mode is extended beyond simple weighted averaging, by using up to five predefined weights (Generalized Bi-prediction (GBI)) and a pixel level motion refinement (Bi-Directional Optical Flow (BDOF)) may be performed on top of it. In order to increase the accuracy of the MVs of the merge mode, a refined operation may be performed around the initial MVs in both reference picture lists L0 and L1 using the Decoder side Motion Vector Refinement (DMVR). Motion vectors are stored at 1/16th-Luma-sample precision for Luma. In addition, the Adaptive Motion Vector Resolution (AMVR) allows the Motion Vector Difference (MVD) of the CU to be coded in one of the three resolutions: Quarter-luma-sample, Integer-luma-sample, and Four-luma-sample (or 1/16 luma-sample in AFF).

Finally, the VTM5.0 inter coder introduces these new concepts: (1) the Triangular prediction (Triang) in which a CU may be further split into two triangular units, in either diagonal or inverse diagonal direction. Each of the two units is predicted using its own Uni-directional MV; (2) Combined Inter and Intra Prediction (CIIP) is proposed to improve the Intra mode in inter pictures, by combining the decided Intra mode with an extra merge indexed prediction; (3) Merge with MVD scheme (MMVD) is used for skip and merge modes with a new motion vector expression method with simplified signaling: The expression method includes starting point, motion magnitude, and motion direction; (4) Symmetric MVD (SMVD), which derives the MVD of reference list 1 from reference list 0, based on the assumption of linear motion in bi-prediction mode.

3.4. Quantization

The maximum Quantization Parameter (QP) is extended from 51 to 63, and a new concept of quantization is introduced: The Dependent Quantization (DepQuant), in which the reconstruction value for a transform coefficient depends on the value of the transform coefficient that precedes it in the reconstruction order.

3.5. Transform

Large block-size transforms, of up to 64x64 pixels, are used. High-frequency transform coefficients are zeroed out, so that only the lower-frequency coefficients (top-left 32×32 block) are retained. VTM5.0 uses Enhanced Multiple Transform (EMT), where two new transform matrices are added in addition to DCT-II, namely the DST-VII and the DCT-VIII. Moreover, to reduce the size of the matrices

of transformed coefficients, a Low-Frequency Non-Separable Transform (LFNST) is applied between transform and quantization at encoder and between de-quantization and inverse transform at decoder side.

3.6. In-loop Filtering

Besides deblocking filter and Sample Adaptive Offset (SAO) used in HEVC, the Adaptive Loop Filter (ALF) is applied directly on the reconstructed samples of the SAO process, where one filter among 25 filters is selected for each 4×4 block, based on the direction and activity of local gradients. Nevertheless, the Luma Mapping with Chroma Scaling (LMCS) is performed before the in-loop filtering. This tool adjusts the input luma signal by redistributing it across the dynamic range using a piecewise linear mapping function and scale the chroma residuals according to the average value of the corresponding luma samples. For an inter-predicted CU, the Sub-Block Transform for inter blocks (SBT) may be used instead of EMT to code only a part of the residual block with inferred adaptive transform and the other part of the residual block is zeroed out.

3.7. Entropy Coder

VVC still uses the same entropy coding method used in HEVC (Context-adaptive binary arithmetic coding (CABAC)), but with some changes: The CABAC engine uses a 2-state model with variable probability updating window sizes, instead of the pre-computed LUT of the HEVC. The transform coefficients within a Coefficient Group (CG) are coded according to pre-defined scan orders in five passes. And finally, the selected probability model and binarization models depend on the local neighborhood, where the template used to specify the local neighborhood is defined by the 5 nearby samples in the left-bottom of the current coefficient. For more information about VVC CABAC, refer to [11].

VVC is mainly targeting 4K and 8K video resolutions. The added coding tools compared to HEVC imply new syntax elements in the bitstream, including flags. While this is not an issue at high resolutions and high-bitrates, it may represent an unnecessary overhead at low resolutions and low-bitrates.

4. PROPOSED METHODOLOGY

Our aim in what follows is to identify the subset of coding tools that may be disabled in low-resolution and low-bitrate use cases, to provide a significant reduction in terms of coding complexity, while preserving compression efficiency. This can be formulated as a constrained optimization problem, which is solved using a branch-and-prune approach. This technique identifies the individual tools and combination thereof that may be safely disabled, and those that have to be kept activated.

4.1. Problem Formulation

Consider a set of video sequences $\mathcal{V} = \{v_1, \ldots, v_N\}$. The performance of a video encoder can be measured by the rate R (in Kbps) required to store the compressed videos, the resulting distortion D of the decoded videos (typically measured using the weighted average *PSNR* of the three components *Y*, *U*, and *V* [3]), and the complexity C of the encoding process (approximated by the run-time, measured in seconds). The values of the triple (R, D, C) depend on the input video sequence v_n and on the *encoding parameter vector* $\mathbf{p} = \left(p_1, \ldots, p_{n_\mathrm{p}}\right)$ of the video coder as follows

$$(R, D, C) = f\left(v_n, \mathbf{p}\right), \tag{1}$$

where f is some (unknown) nonlinear function describing the behavior of the considered video coder. The components of $\mathbf{p}$ represent the coder input parameters, which may be adjusted to get different trade-offs between R, D, and C. The parameter vector $\mathbf{p}$ may be partitioned into subvectors. One may identify:

- $\mathbf{p_T}$ representing binary-valued parameters indicating whether some tools are activated or remain unused;

- $\mathbf{p_C}$ representing a finite-valued of configuration inputs for the preceding tools, *e.g.*, the *TargetBitrate* and *InitialQP* must be specified for the Rate Control, both are integer values;

- $\mathbf{p_O}$ corresponding to other parameters which do not belong to any tool, *e.g.*, *GOP size* and *GOP type* configurations.

To properly evaluate the performance of a coding tool, several target values of the rate R have to be considered. In our work, we use the *Bjontegaard Delta Rate* ($\mathrm{BD_{rate}}$) [12] to evaluate the loss of a set $\mathcal{P}_1 = \{\mathbf{p}_1^{(1)}, \ldots, \mathbf{p}_1^{(n_\mathrm{DR})}\}$ compared to another set $\mathcal{P}_2 = \{\mathbf{p}_2^{(1)}, \ldots, \mathbf{p}_2^{(n_\mathrm{DR})}\}$ of values of the parameter vector. The vectors $\mathbf{p}_j^{(i)} \in \mathcal{P}_j$, $j = 1, 2$ share the same components $\mathbf{p}_{\mathrm{T},j}$, $\mathbf{p}_{\mathrm{C},j}$, and $\mathbf{p}_{\mathrm{O},j}$, but take distinct QP values $\mathrm{QP}^{(i)}$, $i = 1, \ldots, n_\mathrm{DR}$, with $n_\mathrm{DR} \geq 4$. Sets of parameter vectors $\mathcal{P}_j$ are called *parameter configuration sets* (PCS) in what follows.

Consider some reference PCS $\overline{\mathcal{P}}$, corresponding, *e.g.*, to the best rate-distortion compromise for a set of video sequences. Our aim is to find a PCS $\underline{\mathcal{P}}$ such that

$$\underline{\mathcal{P}} = \arg\min_{\mathcal{P}} C\left(\mathcal{P}\right) \tag{2}$$

$$\text{such that } \mathrm{BD_{rate}}(v, \overline{\mathcal{P}}, \mathcal{P}) \leqslant \Delta_\mathrm{rate}, \tag{3}$$

$$\left(R^{(i)}, D^{(i)}, C^{(i)}\right) = f\left(v, \mathbf{p}^{(i)}\right), \mathbf{p}^{(i)} \in \mathcal{P}$$

where $\Delta_\mathrm{rate} > 0$ is the largest tolerated loss in terms of $\mathrm{BD_{rate}}$ and $C\left(\mathcal{P}\right) = \sum_{i=1}^{n_\mathrm{DR}} C^{(i)}$. $\underline{\mathcal{P}}$ is a PCS minimizing the complexity, while keeping good compression performance compared to the optimal parameter set.

4.2. Search for a good Parameter Configuration Set

In what follows, we propose a method to solve the optimization problem (2) in an approximate way. Our approach concentrates on finding the subvector $\mathbf{p}_{\mathrm{T},j}$ indicating the activated and disabled tools.

Consider $\overline{\mathbf{p}}_\mathrm{T}^{(i)} \in \overline{\mathcal{P}}$, the parameter vectors indicating the set of tools activated in the reference PCS. First, one builds all candidate PCS $\mathcal{P}_{1,j} = \{\mathbf{p}_j^{(1)}, \ldots, \mathbf{p}_j^{(n_\mathrm{DR})}\}$, $j = 1, \ldots, n_1$ with subvectors $\mathbf{p}_{\mathrm{T},j}^{(i)}$ obtained by disabling a *single* tool activated in $\overline{\mathbf{p}}_\mathrm{T}^{(i)}$, *i.e.*, $d_\mathrm{H}(\mathbf{p}_{\mathrm{T},j}^{(i)}, \overline{\mathbf{p}}_\mathrm{T}^{(i)}) = 1$, where d_H is the Hamming distance. Only the candidate PCS such that (3) is satisfied are further considered, the others are pruned.

Second, assuming that $n_1' \leqslant n_1$ PCS satisfy (3), these PCS are sorted. PCS with gains in terms of $\mathrm{BD_{rate}}$ are sorted first, and then PCS with a good complexity reduction and a small $\mathrm{BD_{rate}}$ loss. Let $\mathbb{P}_1 = \left\{\mathcal{P}_{1,1}', \ldots, \mathcal{P}_{1,n_1'}'\right\}$ be the ordered set of these PCS. The PCS providing a gain in terms of $\mathrm{BD_{rate}}$ (negative $\mathrm{BD_{rate}}$) are ordered first in $\mathbb{P}_1$ by decreasing $\mathrm{BD_{rate}}$ gain. Then, the PCS providing a $\mathrm{BD_{rate}}$ loss (positive $\mathrm{BD_{rate}}$) are ordered by decreasing value of

$$\lambda_j = \frac{\left(C\left(\overline{\mathcal{P}}\right) - C\left(\mathcal{P}_{1,j}\right)\right) / C\left(\overline{\mathcal{P}}\right)}{\mathrm{BD_{rate}}(v, \overline{\mathcal{P}}, \mathcal{P}_{1,j})},$$

where the numerator of λ_j is the relative complexity decrease provided by the PCS $\mathcal{P}_{1,j}$.

Third, the set $\mathbb{P}_1$ is split into two parts $\mathbb{P}_2$, containing the $n_2 \leqslant n_1'$ first elements of $\mathbb{P}_1$ and $\mathbb{P}_3$ containing the remaining elements. The set $\mathbb{P}_2$ contains the most promising candidates PCS with a single tool disabled compared to $\overline{\mathcal{P}}$. The greedy approach presented in Algorithm 1 is then used to combine candidate PCS, *i.e.*, disable more tools, while satisfying (3). In Algorithm 1, assuming that $\mathcal{P}_1 = \{\mathbf{p}_1^{(1)}, \ldots, \mathbf{p}_1^{(n_{\text{DR}})}\}$ and $\mathcal{P}_2 = \{\mathbf{p}_2^{(1)}, \ldots, \mathbf{p}_2^{(n_{\text{DR}})}\}$, the notation $\mathcal{P}_1 \wedge \mathcal{P}_2$ corresponds to the PCS $\mathcal{P}_3 = \mathcal{P}_1 \wedge \mathcal{P}_2 = \{\mathbf{p}_3^{(1)}, \ldots, \mathbf{p}_3^{(n_{\text{DR}})}\}$ such that $\mathbf{p}_{\text{T},3}^{(i)} = \mathbf{p}_{\text{T},1}^{(i)} \wedge \mathbf{p}_{\text{T},2}^{(i)}$.

Input: $\mathbb{P}_2$
Output: $\mathcal{P}_1$
Initialization: extract $\mathcal{P}_1$, the first element of $\mathbb{P}_2$;
while $\mathbb{P}_2 \neq \emptyset$ **do**
  Extract $\mathcal{P}_2$, the next element of $\mathbb{P}_2$;
  if $C(\mathcal{P}_1 \wedge \mathcal{P}_2) \leqslant C(\mathcal{P}_1)$ *and* $BD_{rate}(v, \overline{\mathcal{P}}, \mathcal{P}_1 \wedge \mathcal{P}_2) \leqslant \Delta_{rate}$ **then**
    $\mathcal{P}_1 = \mathcal{P}_1 \wedge \mathcal{P}_2$;
  end
end

Algorithm 1: Evaluating the best PCS

Algorithm 1 progressively disable tools corresponding to the PCS in $\mathbb{P}_2$, starting with the most promising PCS. When disabling a tool results in a complexity reduction while satisfying (3), the PCS is updated. Tools, when disabled, do not reduce the complexity or lead to a large loss in BD_{rate} are kept activated.

Finally, a branch-and-prune approach presented in Algorithm 2 is considered, starting from the PCS $\mathcal{P}_1$ provided by Algorithm 1 to select additional tools to disable corresponding to PCS in $\mathbb{P}_3$. One tries first to disable a single additional tool from $\mathcal{P}_1$ corresponding to the various PCS in $\mathbb{P}_3$. All PCS $\mathcal{P} \in \mathbb{P}_3$ such that $\mathcal{P}_1 \wedge \mathcal{P}$ leading to a performance decrease compared to $\mathcal{P}_1$ are discarded from $\mathbb{P}_3$. Then pairs, triples, *etc*. of PCS remaining in $\mathbb{P}_3$ are considered.

Input: $\mathcal{P}_1$ and $\mathbb{P}_3$
Output: $\mathcal{P}$
Initialization: $i = 1$;
while *number of PCSs in* $\mathbb{P}_3 > 1$ **do**
  Build $\mathbb{P}$, all combinations of i tools from $\mathbb{P}_3$;
  while $\mathbb{P} \neq \emptyset$ **do**
    Extract $\mathcal{P}_2$, first PCSs in $\mathbb{P}$;
    if $C(\mathcal{P}_1 \wedge \mathcal{P}_2) \geqslant C(\mathcal{P}_1)$ *and* $BD_{rate}(v, \overline{\mathcal{P}}, \mathcal{P}_1 \wedge \mathcal{P}_2) \geqslant \Delta_{rate}$ **then**
      Discard $\mathcal{P}_2$ from $\mathbb{P}$;
    end
  end
  Put all PCSs of $\mathbb{P}$ in $\mathbb{P}_3$;
  $i = i + 1$;
end
Extract $\mathcal{P}_2$, the only PCS in $\mathbb{P}_3$;
$\mathcal{P} = \mathcal{P}_1 \wedge \mathcal{P}_2$;

Algorithm 2: Branch-and-prune method

Let $\mathcal{P}_2$ the PCS (or combination of PCS) in $\mathbb{P}_3$ leading to the smallest value of $C(\mathcal{P}_1 \wedge \mathcal{P}_2)$ while $\text{BD}_{\text{rate}}(v, \overline{\mathcal{P}}, \mathcal{P}_1 \wedge \mathcal{P}_2) \leqslant \Delta_{\text{rate}}$. Then the PCS $\mathcal{P} = \mathcal{P}_1 \wedge \mathcal{P}_2$ is an approximate solution of (2).

5. PERFORMANCE EVALUATION

5.1. Experimental setup

We selected 14 JVET test sequences defined in the Common Test Conditions (CTC) [13], each one of the sequences has at most 300 images. In a first phase, 7 sequences were considered to apply our approach and identify the best PCSs. Then, in a second phase, tests are conducted on all sequences to evaluate the performance obtained with the previously identified best PCSs.

All sequences have been temporally sub-sampled at 30 fps and spatially sub-sampled using FFmpeg [14] resulting in frames of 384×216, 512×288, and 640×360 pixels. A Random Access (RA) configuration is selected according to JVET CTC [13] and QP values are chosen in $\{27, 32, 37, 42\}$. VTM5.0 is used in the experiments and run on a PC with 2 Intel Xeon CPU E5-2670 v3 24 cores @ 2.30 GHz running under Linux. The threshold Δ_{rate} is fixed to 2% as we have noticed that this loss is subjectively unnoticeable. The value of n_2 helps to get a trade-off between complexity and accuracy in the search for $\underline{\mathcal{P}}$. Here, we take $n_2 = n_1'/2$. The new tools of VTM5.0 we have tested in this work are listed in Table 1.

DualTree	Separate Partitioning for Luma & Chroma in I-slice
CCLM	Chroma prediction based on linear model
MRL	Multiple Reference Line intra prediction
MIP	Matrix-based Intra prediction
ISP	Intra Sub-Partitions
CIIP	Combined Inter and Intra prediction
SbTMVP	Sub-Pu Temporal Motion Vector Prediction
AFF	AFFine inter motion compensation
MMVD	Merge with MVD
SMVD	Symmetric MVD
Triang	inter predictions for Triangular Units
GBI	Generalized Bi-prediction
BDOF	Bi-Directional Optical Flow
DMVR	Decoder Side Motion Vector Refinement
AMVR	Adaptive MV Resolution
EMT	Enhanced Multiple Transform
LFNST	Low-Frequency Non-Separable Transform
SBT	Sub-Block Transform for inter blocks
LMCS	Luma Mapping with Chroma Scaling
ALF	Adaptive Loop Filter

Table 1. New tools of VTM5.0 tested in this work

5.2. Analysis

In this section, we present experimental results in order to illustrate the proposed approach. Table 2 presents the detailed BD_{rate} and complexity reduction C when disabling one tool at a time for the *Johnny* sequence of resolutions 384×216, 512×288 and 640×360. These percentages are calculated relative to VTM5.0 with all tools activated (a negative BD_{rate} indicates a gain with respect to VTM5.0).

From Table 2, one observes that disabling tools related to inter-frame coding (*e.g.*, AFF, MMVD, and Triang in resolutions 384×216) and inloop filtering (ALF, LMCS) leads to significant gains in complexity. Other tools related to transform operations such as LFNST and EMT also lead to a significant complexity decrease. Disabling a tool can sometimes lead to an improved BD_{rate} when operating at low resolutions and low-bitrates, such as LMCS, SMVD

Johnny at 384x216				Johnny at 512x288				Johnny at 640x360			
Disabled Tools	$BD_{\text{rate}}\%$	$C\%$	λ	Disabled Tools	$BD_{\text{rate}}\%$	$C\%$	λ	Disabled Tools	$BD_{\text{rate}}\%$	$C\%$	λ
LMCS	-0.42	10.77	-	LMCS	-0.40	13.28	-	BDOF	-0.29	13.20	-
SMVD	-0.01	8.30	-	SMVD	0.02	8.98	422.56	SMVD	-0.08	9.67	-
AMVR	-0.01	8.46	-	MMVD	0.15	15.56	104.76	LMCS	-0.07	9.85	-
GBI	0.01	8.74	1544.07	BDOF	0.14	10.83	77.86	MMVD	0.10	15.67	161.51
CIIP	0.03	7.98	271.58	ISP	0.37	20.55	55.37	CIIP	0.06	7.65	121.65
MMVD	0.20	14.21	72.12	CIIP	0.14	7.46	52.16	AMVR	0.15	10.06	66.67
AFF	0.29	16.59	57.98	AFF	0.58	19.87	34.48	AFF	0.58	18.72	32.06
MIP	0.14	8.30	57.41	AMVR	0.26	8.60	33.43	Triang	0.41	11.39	28.00
Triang	0.28	10.04	36.47	Triang	0.39	11.69	29.77	MIP	0.45	9.30	20.75
MRL	0.20	7.23	36.01	SBT	0.31	6.72	21.92	SbTMVP	0.28	4.54	16.50
SbTMVP	0.14	4.81	34.35	MIP	0.45	7.81	17.32	SBT	0.24	3.80	15.60
SBT	0.07	1.64	24.45	MRL	0.39	6.38	16.27	EMT	0.42	6.05	14.49
EMT	0.34	7.01	20.77	SbTMVP	0.32	5.08	16.04	GBI	0.69	9.71	14.04
ALF	1.60	28.15	17.55	GBI	0.78	9.72	12.45	ISP	0.49	5.88	11.94
LFNST	0.87	12.30	14.12	LFNST	1.07	12.89	12.03	LFNST	1.36	12.69	9.33
BDOF	0.76	9.56	12.65	EMT	0.51	5.89	11.45	ALF	2.11	18.04	8.56
ISP	0.55	6.35	11.59	ALF	1.98	20.55	10.38	MRL	0.74	5.81	7.87
DMVR	0.59	5.30	8.95	CCLM	0.82	6.28	7.70	CCLM	0.88	5.46	6.18
CCLM	0.79	5.94	7.50	DMVR	0.91	4.35	4.76	DMVR	0.78	4.48	5.72
DualTree	0.29	-2.29	-7.94	DualTree	0.32	-1.71	-5.37	DualTree	0.70	-3.00	-4.32

Table 2. BD_{rate} and complexity reduction C in "Johnny" test sequence when disabling one tool at time

Resolution	Video	R Kbps	$BD_{\text{rate}}\%$	$C\%$	Disabled Tools
384x216	Johnny	18-72	1.88	56.06	**LMCS GBI LFNST MMVD MIP CIIP** SMVD MRL SBT Triang AFF AMVR SbTMVP
	Basketball	48-368	1.97	37.07	**LMCS GBI LFNST EMT MIP CIIP** SMVD
	DaylightRoad2	48-367	2.00	48.91	**LMCS GBI LFNST MMVD EMT MIP CIIP** MRL SBT Triang
	BQMall	50-331	1.98	44.93	**LMCS GBI MMVD EMT MIP CIIP** SMVD AFF CCLM
	Drums	89-538	1.74	43.27	**LMCS GBI LFNST EMT MIP CIIP** SMVD MRL AFF SbTMVP
	RaceHorses	50-403	2.00	38.82	**LMCS GBI LFNST MMVD EMT** MRL SBT
	Kimono	27-271	1.91	51.21	**LMCS GBI LFNST MMVD EMT** SMVD MRL AFF ISP
Average	-	-	1.93	45.75	-
512x288	Johnny	23-102	1.97	57.01	**LMCS CIIP MMVD** AFF Triang SBT ISP BIO AMVR
	Basketball	75-576	1.81	35.25	**LMCS CIIP EMT LFNST SMVD** MRL MIP SBT
	DaylightRoad2	75-575	1.88	48.37	**LMCS CIIP EMT LFNST MMVD** MRL MIP GBI Triang SBT
	BQMall	71-477	1.82	34.92	**LMCS CIIP EMT MMVD SMVD** MRL GBI CCLM
	Drums	125-661	2.00	35.25	**CIIP EMT LFNST SMVD** AFF CCLM SbTMVP
	RaceHorses	78-763	1.79	34.34	**LMCS CIIP EMT LFNST SMVD** AFF CCLM SbTMVP
	Kimono	41-427	1.84	47.78	**LMCS CIIP EMT LFNST MMVD SMVD** MRL ISP AFF IMV
Average	-	-	1.87	41.85	-
640x360	Johnny	30-148	1.92	55.24	**LMCS MIP SMVD CIIP MMVD** AFF Triang AMVR BIO
	Basketball	99-700	1.89	32.45	**LMCS MIP SMVD EMT** LFNST SbTMVP
	DaylightRoad2	100-702	1.89	41.68	**LMCS MIP SMVD MRL GBI EMT MMVD** LFNST SBT
	BQMall	95-639	2.00	36.99	**LMCS MIP SMVD MRL GBI EMT MMVD** CCLM
	Drums	164-789	1.83	41.26	**LMCS MIP SMVD MRL GBI CIIP** AFF SbTMVP
	RaceHorses	109-631	1.30	40.17	**LMCS MIP SMVD MRL GBI EMT CIIP MMVD** SBT ISP
	Kimono	54-531	1.80	45.77	**LMCS SMVD MRL GBI CIIP** LFNST AFF Triang
Average	-	-	1.80	41.94	-

Table 3. BD_{rate} and complexity reduction C of best PCS for tested resolutions and sequences; Selected common tools are in bold

	384x216		512x288		640x360	
Common Tools	LMCS GBI LFNST MMVD EMT MIP CIIP		LMCS CIIP MMVD SMVD EMT LFNST		LMCS MIP CIIP EMT SMVD MRL GBI MMVD	
Video	$BD_{\text{rate}}\%$	$C\%$	$BD_{\text{rate}}\%$	$C\%$	$BD_{\text{rate}}\%$	$C\%$
Johnny	2.34	38.76	2.13	39.13	0.99	42.01
Basketball	2.00	37.95	1.83	24.57	1.57	26.03
DaylightRoad2	0.68	38.96	0.94	35.26	0.98	35.44
BQMall	1.24	37.81	1.59	35.73	1.33	35.31
Drums	1.87	37.94	1.63	34.12	1.21	35.12
RaceHorses	1.87	35.65	1.56	33.96	0.33	26.45
Kimono	0.94	41.60	1.83	38.24	2.39	35.47
ParkScene	0.90	42.25	0.77	41.26	0.85	42.71
KristenAndSara	1.99	38.62	1.87	38.93	0.96	39.32
CatRobot	1.42	42.58	1.41	37.39	1.27	23.79
Tango	1.20	38.55	1.58	36.06	0.95	35.18
ToddlerFountain	1.20	38.55	1.58	36.06	1.43	39.89
SlideShow	1.52	27.20	1.44	23.61	2.88	23.03
SlideEditing	1.09	41.68	0.63	39.72	1.64	40.25
Average	1.45	38.44	1.49	35.29	1.34	34.29

Table 4. BD_{rate} and complexity reduction C when disabling the common combinations of tools. Highlighted cells do not satisfy (3).

and AMVR.

Table 3 shows the BD_{rate} and complexity reduction C for the best combination of disabled tools as obtained by the method described in Section 4.2 for seven test sequences and three resolutions. When several tools are disabled, the complexity gains accumulate in most of the cases. Nevertheless, this observation does not hold for

BD_{rate}, due to the complex dependency among tools and their influence on the coding efficiency. Considering *Johnny* at the resolution of 384×216, jointly disabling the tools: { LMCS GBI LFNST MMVD MIP CIIP SMVD MRL SBT Triang AFF AMVR SbTMVP }, leads to a complexity reduction of 56.06%, with a BD_{rate} loss of 1.88%. For resolutions of 512×288 and 640×360, a complexity reduction of 57.01% and 55.24% is achieved with a BD_{rate} loss of 1.97% and 1.92% respectively. Similar results are obtained when applying the same methodology on other sequences such as, *BasketballDrive*, *DaylightRoad2*, *BQMall*, *Drums*, *RaceHorses* and *Kimono*.

These results are obtained by merely disabling tools and without algorithmic optimization. Nevertheless, the combination of tools that provide the optimal complexity reduction is different from one sequence to the other and even from one resolution to the other. The amount of complexity reduction is also varying. This is due to the spatial and temporal properties of sequences.

Yet, for each resolution, using the results of Table 3, it is possible to identify one common combination of tools satisfying constraint (3) for all sequences. Accordingly, these PCSs are: {*LMCS GBI LFNST MMVD EMT MIP CIIP*} for resolution 384×216, {*LMCS CIIP MMVD SMVD EMT LFNST*} for 512×288, and {*LMCS MIP CIIP EMT SMVD MRL GBI MMVD*} for 640×360. Table 4 shows the BD_{rate} and complexity reduction C for the previously identified PCSs considering the 14 test sequences. We observe that the constraint (3) is satisfied in most cases. Thus, putting these PCSs in separate profiles for each resolution will be beneficial for use cases with real-time and low-bitrate constraints. We conclude that the PCS identification approach presented in this paper provides results that are likely to be generalized to a larger set of video sequences.

6. CONCLUSIONS

This paper proposes an optimization of the next generation VVC targeting low-resolution video sequences encoded at low-bitrates (less than 700 kbps). The aim is to identify a set of coding tools which may be disabled while preserving coding efficiency. For this purpose, a branch-and-prune approach is proposed to determine the set of coding tools which provide the best complexity reduction, while satisfying a constraint on the BD_{rate} degradation.

Experimental results show that significant reduction of encoding complexity can be achieved, with negligible BD_{rate} loss. For instance, the joint disabling of the tools { LMCS MMVD AFF MIP Triang SMVD AMVR GBI CIIP MRL SbTMVP SBT LFNST } leads to a complexity reduction of 56%, with a loss of 1.88% in BD_{rate}, for *Johnny* at a 384×216 resolution. Moreover, a common combination of tools to disable for each resolution was determined. Our experimental results show that these common PCSs most likely cause less than 2% BD_{rate} loss with good reduction in terms of encoding complexity, which is beneficial for use cases with real-time and low-bitrate constraints.

7. REFERENCES

[1] VNI Cisco, "Cisco visual networking index: Forecast and trends, 2017–2022," *White Paper*, 2018.

[2] G. Sullivan, "Deployment status of the HEVC standard," document jctvc-ab0020 of jct-vc,, JCT-VC, Torino, Italy, 2017.

[3] T. Grois D., Nguyen and Marpe D., "Performance comparison of av1, jem, vp9, and hevc encoders," in *Applications of Digital Image Processing XL*. International Society for Optics and Photonics, 2018, vol. 10396, p. 103960L.

[4] Convenor of MPEG, "N17482: Versatile video coding project starts strongly in the joint video experts team," *MPEG Press Release*, 2018.

[5] X. Zhang W. Zhao L. Shen, Z. Liu and Z. Zhang, "An effective cu size decision method for hevc encoders," *IEEE transactions*, 2012.

[6] S. Ma L. Zhao, L. Zhang and D. Zhao, "Fast mode decision algorithm for intra prediction in hevc," in *2011 Visual Communications and Image Processing (VCIP)*. IEEE, 2011, pp. 1–4.

[7] T. L. Da Silva, L. V. Agostini, and L. A. da Silva Cruz, "Fast HEVC intra prediction mode decision based on EDGE direction information," in *Proc. European Signal Processing Conference*, 2012.

[8] F. Sampaio, S. Bampi, M. Grellert, L. Agostini, and J. Mattos, "Motion vectors merging: low complexity prediction unit decision heuristic for the inter-prediction of HEVC encoders," in *Proc. IEEE International Conference on Multimedia and Expo*. IEEE, 2012, pp. 657–662.

[9] Joint Video Exploration Team, "Versatile video coding (vvc)," https://jvet.hhi.fraunhofer.de/, 2018.

[10] Convenor of JVET, "N15340: Requirements for a future video coding standard v1," in *MPEG 112th, Warsaw, Poland*, 2015.

[11] Y. Ye J. Chen and S. Kim, "Jvet-n1002-v1: Algorithm description for versatile video coding and test model 5 (vtm 5)," in *Jvet 14th Meeting: Geneva, CH, 19 27 Mar. 2019*.

[12] P. Hanhart and T. Ebrahimi, "Calculation of average coding efficiency based on subjective quality scores," *Journal of Visual Communication and Image Representation*, vol. 25, no. 3, pp. 555–564, 2014.

[13] Teruhiko Suzuki, "Jvet-d1002 : Work plan for assessment of test material," *JVET 4th Meeting, Chengdu, CN.*, 2016.

[14] FFmpeg Developers, "ffmpeg tool (build: ffmpeg-20180716-8aa6d9a-win64-static)," http://ffmpeg.org/, 2019.