Design Space Exploration of Practical VVC Encoding for Emerging Media Applications

Joose Sainio, *Member, IEEE*, Alexandre Mercat, *Member, IEEE*, and Jarno Vanne, *Member, IEEE*

***Abstract*—Versatile Video Coding (VVC/H.266) is the latest video coding standard designed for a broad range of next-generation media applications. This paper explores the design space of practical VVC encoding by profiling the Fraunhofer Versatile Video Encoder (VVenC). All experiments were conducted over five 2160p video sequences and their downsampled versions under the random access (RA) condition. The exploration was performed by analyzing the rate-distortion-complexity (RDC) of the VVC block structure and coding tools. First, VVenC was profiled to provide a breakdown of coding block distribution and coding tool utilization in it. Then, the usefulness of each VVC coding tool was analyzed for its individual impact on overall RDC performance. Finally, our findings were elevated to practical implementation guidelines: the highest coding gains come with the multi type tree (MTT) structure, adaptive loop filter (ALF), cross component linear model (CCLM), and bi-directional optical flow (BDOF) coding tools, whereas multi transform selection (MTS) and affine motion estimation are the primary candidates for complexity reduction. To the best of our knowledge, this is the first work to provide a comprehensive RDC analysis for practical VVC encoding. It can serve as a basis for practical VVC encoder implementation or optimization on various computing platforms.**

***Index Terms*—Coding tree unit (CTU) structure, design space exploration (DSE), rate-distortion-complexity (RDC), video coding, versatile video coding (VVC).**

I. Introduction

VIDEO is ubiquitous in our everyday life and paves the way for digitalized social interaction. The powerful devices for rich media creation and consumption with always online philosophy have created a fertile ground for a plethora of media applications that fuel the exponential growth of video traffic. Cisco has estimated that 82% of all IP traffic is video in 2022 [1].

To deal with the ever-increasing video volume, *Joint Video Experts Team* (*JVET*), formed by MPEG and ITU-T, has published a series of video coding standards. *Versatile Video Coding* (*VVC*) [2] is the latest one of these standards. It was introduced as successor to the widespread *High Efficiency Video Coding* (*HEVC*) [3]. VVC is able to improve coding efficiency for the same objective visual quality by more than 30% over HEVC [4] and the gap widens to 40% with subjective quality [5]. However, this quality increase comes at a cost of over eight times encoding complexity under the *random access* (*RA*) condition [4] of the *common test conditions* (*CTC*) [6].

The complexity and quality improvement of VVC primarily stems from its new *quadtree + multi-type tree* (*QT+MTT*) block partitioning scheme [7], [8]. In addition, there are many new coding tools in each coding stage. These stages are depicted in Fig. 1 and they include *intra prediction* (*IP*), *motion estimation and compensation* (*ME/MC*) a.k.a. inter prediction, *forward/inverse transform and quantization* (*TR/Q*), *entropy coding* (*EC*), and *loop filtering* (*LF*).

In VVC, the frames are split into *coding tree units* (*CTUs*) of maximum size of 128 × 128 luma samples. CTUs are further partitioned into *coding units* (*CUs*) with the QT+MTT. QT divides a CU into four identical square CUs, whereas the MTT can either use *binary tree* (*BT*) or *ternary tree* (*TT*) splits, which can result in both rectangular and square CUs. Fig. 2(a) depicts all these partitioning and Fig. 2(b) exemplifies a CTU split with different types of QT+MTT partitions. Overall, the QT+MTT can divide the CTU in thousands of different possible ways that need to be checked with all the tools for optimal encoding, which causes the massive increase in complexity.

The current technology advancements also lead the transition from stationary multimedia workstations to more resource-constrained smartphones and other handheld consumer devices [9]. For them, efficient codec implementations are of utmost importance in order to tackle computational complexity of VVC with acceptable coding efficiency and power budget.

Currently, there are two noteworthy open-source VVC encoders: the *VVC test model* (*VTM*) [10] and the *Fraunhofer Versatile Video Encoder* (*VVenC*) [11], [12]. VTM is the reference encoder, maintained by JVET. It implements all VVC coding tools, but it is poorly optimized and far from practical use. VVenC is optimized from VTM and designed for practical VVC encoding, but it is still unable to reach real-time speed.

Table I compares the most prominent VVC complexity analyses [4], [8], [13]–[21] with our work. The comparison includes the profiled encoder(s), the conditions, as well as the characterization of complexity, *rate distortion* (*RD*), and QT+MTT analyses. In addition, it reports the main novelty aspects of each analysis.

Manuscript received 1 October 2021; revised 20 January 2022, 4 March 2022 and 16 May 2022; accepted 7 July 2022. Date of publication 28 July 2022; date of current version 24 October 2022. This work was supported in part by the AI for Situational Awareness (AISA) Project led by Nokia and funded by Business Finland. *(Corresponding author: Joose Sainio.)*

The authors are with the Ultra Video Group, Tampere University, 33101 Tampere, Finland (e-mail: joose.sainio@tuni.fi; alexandre.mercat@tuni.fi; jarno.vanne@tuni.fi).

Digital Object Identifier 10.1109/TCE.2022.3194596

This work is licensed under a Creative Commons Attribution 4.0 License. For more information, see https://creativecommons.org/licenses/by/4.0/

TABLE I
CHARACTERIZATION OF THE EXISTING AND CURRENT VVC COMPLEXITY ANALYSES

Work	Encoder	Condition	Complexity analysis				RD analysis				QT+MTT analysis	Main novelty
			overall	stages	tool	other	overall	tool	other	quality metrics		
[13]	VTM2.0	RA	×				×			PSNR		RD comparison of HEVC, VVC and AV1.
[14]	VTM4.0	AI/RA	×				×			PSNR		RD comparison of seven different encoders.
[15]	VTM2.0-13.2*	AI/LD/RA	×		×		×	×		PSNR		Evaluation of the VVC standard.
[16]	VTM8.0	LD/RA	×			memory						Memory profiling of VTM.
[17]	VTM6.0	AI/LD RA	×	×								Complexity breakdown of encoding categories.
[18]	VTM5.0	RA	×			SIMD	×			PSNR		Evaluation of SIMD in VTM.
[4]	VTM10.0*	AI/LD/RA	×	×			×			PSNR/ SSIM/ VMAF		RDC using CTC + 8 UHD sequences and cycle accurate complexity analysis.
[8]	VTM7.0	RA	×		×		×	×		PSNR	RD and complexity.	RDC analysis of tools and QT/BT/TT depth.
[19]	VTM10.0/ VVenC 0.3.1*	AI/LD/RA	×			SIMD	×			PSNR		Evaluation of VVC decoding and VVenC.
[20]	VTM7.0	AI	×		×						Average usage of CU sizes.	Analysis of CTU partitioning for the AI condition.
[21]	VTM3.0	AI	×		×							Evaluation of complexity reduction opportunities for AI coding.
Proposed	**VVenC 0.3.1***	**RA**			×	**Res.****	×	×	**Res.****	**PSNR/ SSIM/ VMAF**	**Average usage of CU sizes. Usage of tools for CU sizes.**	**RDC analysis of tools and overall analysis of tool usage and CTU structure as a function of resolution and coding characteristics of sequence.**

* Using standardized version of the encoder. **Resolution.

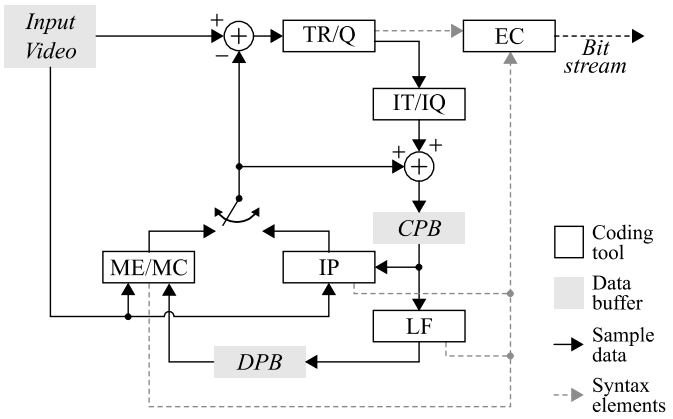


Fig. 1. Simplified block diagram of a VVC encoder.

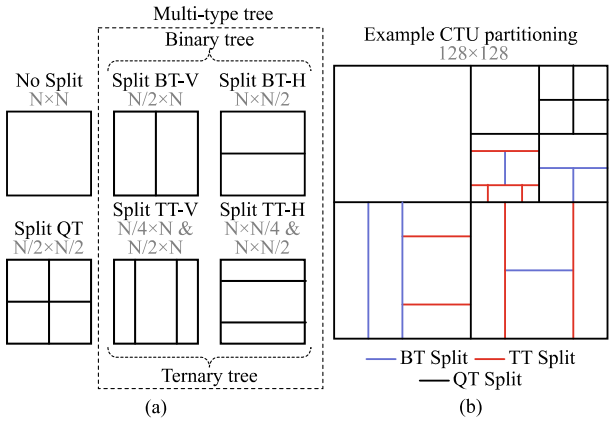


Fig. 2. CTU partitioning in VVC. (a) VVC split types. (b) Example.

García-Lucas *et al.* [13] and Laude *et al.* [14] only performed overall *rate distortion complexity* (*RDC*) analysis, i.e., they addressed RDC characteristics at encoder level only. Similarly, the comparison made by JVET [15] only included overall RDC analysis. Additionally, all coding tools were individually evaluated on separate JVET documents during the VVC standardization process. However, these evaluations were performed with the VTM versions available at that time, making the results severely outdated particularly at least for the earliest introduced tools. Furthermore, the evaluation results were not collected into a single document.

The other listed works included more in-depth analyses [4], [8], [16]–[19]. The main contribution of Cerveira *et al.* [16] was memory profiling results with the shares of memory accesses per each encoding stage. Pakdaman *et al.* [17] quantified the average shares of different encoding and decoding tool categories of VTM and HM. Siqueira *et al.* [18] evaluated the complexity of VTM with and without the s*ingle instruction multiple data* (*SIMD*) optimizations. A more thorough complexity analysis was also performed to provide the relative execution times for each encoding stage of VTM in comparison with HM.

Our previous study [4] addressed the RDC characteristics of VTM10.0 by using HM16.22 as an anchor. It was the first study that carried out the RD comparison between VTM and HM with three objective quality metrics: PSNR, SSIM [22], and VMAF [23]. The complexity shares of encoder and decoder stages were profiled at function level.

TABLE II
TEST SEQUENCES

Dataset	Original resolution	Sequence	Frame count	Frame-rate	Bit depth
CTC [6]	3840×2160	*DaylightRoad2*	300	60 fps	10
	4096×2160	*Tango*	294	60 fps	10
UVG [35]	3840×2160	*Bosphorus*	300	60 fps	10
		ShakeNDry	150	60 fps	10
		SunBath	600	50 fps	10

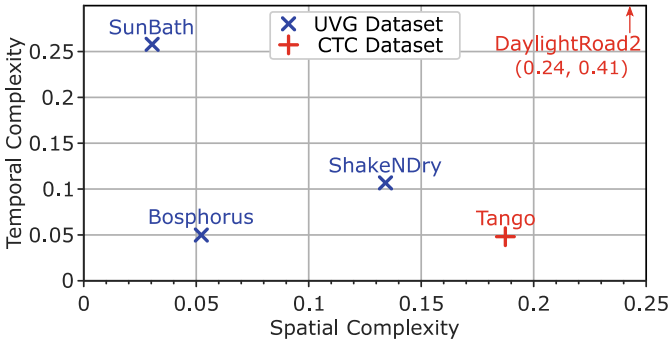


Fig. 3. Complexity scores of the selected sequences.

Brandenburg *et al.* [8] evaluated several VTM encoding tools independently in terms of coding efficiency and complexity. This work also introduced the encoder that would become VVenC [11]. Bossen *et al.* [19] provided a complexity analysis of VTM for individual encoding tools and encoding stages but only at very high level. This study also reported the encoding time distribution of the VVenC encoder.

Finally, Saldanha *et al.* [20] analyzed the encoding partitioning, but only in the *all intra* (*AI*) condition. Tissier *et al.* [21] also only addressed the AI condition when evaluating complexity reduction opportunities of the CTU partitioning, intra mode prediction, and *multiple transforms* (*MTS*) process.

Some other works reported complexity analyses for consumer electronics [24], [25], but only for HEVC. Pescador *et al.* [24] analyzed the complexity of a *digital signal processor* (*DSP*)-based implementation of the HEVC HM9.0 decoder. Engelhardt *et al.* [25] described a complete FPGA implementation of a HEVC decoder that can also be used as a starting point for an ASIC implementation.

This work takes a step forward from the prior art and performs a comprehensive *design space exploration* (*DSE*) that investigates two yet unexplored aspects: 1) the usage of individual VVC tools and their effect on CTU structure; and 2) the effect of resolution on the RD performance at level of coding tools. Only Saldanha *et al.* [20] has considered the CTU structure, but merely in the AI condition. Since the QT+MTT has the highest effect on the RDC, it is important to understand how the encoded content affects the CTU structure and used tools to reduce the complexity of the encoder. Our DSE investigates the composition of the CTU structure in the more commonly used RA condition and the usage of tools with each CU size.

None of the previous analyses consider the effect of resolution on the encoding efficiency. Although VVC is primarily designed for higher resolutions, supporting lower resolutions is still relevant in bandwidth-restricted environments, where VVC can be used to improve the *Quality of Experience* (*QoE*) [26], [27]. Therefore, the test set for our DSE is composed of five 4K sequences and their spatially downsampled versions.

In practice, the DSE is performed by encoding the sequences in the RA condition. First, the analysis focuses on the CTU structure imposed by the new QT+MTT partitioning. Secondly, the usage of VVC tools is investigated per different coding unit (CU) sizes. Finally, we provide a RDC analysis of these tools. To the best of our knowledge, this study is the first to use a practical VVC encoder to 1) analyze the RDC of VVC tools as a function of resolution and content; as well as 2) analyze the CTU structure and tool usage.

Although our analysis is performed on a software encoder, the obtained results can serve as a starting point for implementing and optimizing a hardware VVC encoder as well because they tend to share many RDC characteristics. Respectively, our earlier HEVC complexity study [28] was used as a basis for the implementation of several hardware blocks [29]–[31]. Furthermore, there is a direct link between computational complexity and energy consumption in video coding [32]. With HEVC, it was shown that the energy requirement of the encoding process can be reduced by 80% for minimal visual quality loss [33]. Together, these solidify the usability and usefulness of the proposed analysis for hardware VVC encoder implementations to support its further development on consumer electronic devices.

The rest of this paper is organized as follows. Section II describes the setup for our experiments. Sections III, IV, and V, present the analysis of CTU structure, tool usage per CU size, and rate-distortion-complexity, respectively. Section VI gives implementation guidelines for practical VVC encoders. Finally, Section VII concludes this paper.

II. EXPERIMENTAL SETUP

All our experiments were performed with *uvgVenctester* [34], that is an open-source test automation framework designed for performance and conformance testing of video encoders.

A. Test Sequences

Table II summarizes the selected five test sequences taken from the CTC [6] and UVG [35] datasets. They were selected based on their diverse *spatial* and *temporal encoding complexities*, [35] as illustrated in Fig. 3. *SunBath*, *Bosphorus*, *Tango*, and *DaylightRoad2* represent the complexity extremes and *ShakeNDry* the mid-point. Since both *ShakeNDry* and *Bosphorus* have a native frame rate of 120 *frame per second* (*fps*), they were temporally downsampled to 60 fps for fair comparison.

To evaluate the impact of resolution on the overall complexity, the sequences were also spatially downsampled from the original 2160p resolution to 1440p, 1080p, 720p, 480p, and 240p formats by using the bilinear interpolation filter in

TABLE III
SCALED RESOLUTIONS

Resolution	Height	16:9 Width	16:8.125 Width
2160p	2160	3840	4096
1440p	1440	2560	2728
1080p	1080	1920	2048
720p	720	1280	1368
480p	480	832	912
240p	240	416	456

FFmpeg [36]. Table III tabulates the scaled resolutions. The original aspect ratio is maintained in downsampling (*Tango* has a 16:8.125 ratio and the others a 16:9 ratio).

B. Encoder Configuration and Coding Condition

The VVenC encoder [11], version 0.3.1, was chosen because it was the latest version at the beginning of our experiments. The VVenC software was developed from VTM and optimized for performance with SIMD, better search algorithms, and multithreading [12]. It implements nearly all VVC tools, except the *BiPred with CU-level weight during* (*BCW*). VVenC is able to reach the same objective quality as VTM, but with a magnitude smaller encoding time. Thus, our analysis can be generalized to the whole standard.

The experiments were performed under the commonly used RA condition [6], where frames can be coded in an arbitrary order. In the case of VVenC, the frames are separated into groups of 32 and then formed into a pyramid, where the higher-level frames are encoded with better quality for better overall coding efficiency.

C. Evaluated VVC Coding Tools

Table IV tabulates the VVC coding tools included in our analysis. For each tool, the table reports its encoding stage, acronym or abbreviation, a short description, and whether it is included in the CU usage study in Section VI or RDC study in Section VII. The CTU structure study in Section V is carried out with all tools enabled. Please refer to VVC algorithm description [37] and specification [38] by JVET for further information about the tools.

Primarily, our analysis focuses on the tools newly introduced to VVC. However, the prediction types (UniPred, BiPred, and IntraPred) are included as they are a fundamental part of the encoding pipeline. In addition, TSRC is analyzed as it is complimentary to MTS and LFNST. The tools included in the CU usage study are those that are enabled for each CU individually, e.g., DMVR and BDOF are omitted since they are applied always if the CU fulfils conditions for said tool. The tools of the RDC study are limited to those that are specific to the RA condition, except CCLM that is also analyzed due to it high impact on the coding efficiency in the RA condition [8].

D. Quality Metrics

The coding efficiencies of the encoding tools were compared using the well-known *Bjøntegaard delta bitrate* (*BD-rate*) evaluation method [39], [40] that computes average bitrate differences for the same quality. In our analysis, VVenC with all tools enabled was used as an anchor for the BD-rate calculations, so positive values imply coding loss. In practice, the average difference between the RD curves were interpolated per sequence with piecewise cubic interpolation through RD points of four base *quantization parameter* (QP) values: 22, 27, 32, and 37, as defined in the VVC CTC [6]. Altogether, BD-rate was computed with three objective image quality metrics: 1) PSNR, 2) SSIM [22], and 3) VMAF [23].

E. Complexity Profiling Setup

Our complexity profiling was performed on a 22-core general-purpose processor. For reliable results, the number of background programs was minimized.

Ideally, the complexities of individual coding tools could be extracted from encoding runs with all tools enabled. However, this is impossible in practice since some of the VVenC tools are interleaved. Therefore, the complexities are measured by performing the encoding runs with a single tool disabled at a time. This approach also allows us to extract the respective BD-rate values. Because our analysis required over two thousand runs, each run was limited to the first intra period, i.e., only the first 64 frames of each sequence were encoded.

III. ANALYSIS OF VVC CTU STRUCTURE

The largest difference between HEVC and VVC is the MTT that introduce the recursive BT/TT in addition to the recursive QT partitioning included in HEVC. The QT+MTT partitioning scheme combined with the larger CTUs increases the number of different CU sizes from four to seventeen. The MTT partitioning is the main source of the increased complexity in VVC, i.e., predicting it perfectly would decrease the total encoding complexity by 97% in the AI condition [21]. It is therefore crucial to analyze the output QT+MTT partitioning of the CTU structure.

Table V tabulates the share of different sized CUs per resolution for each sequence and the overall averages. For each non-square CU, the shares of wide and tall CUs are combined into one, i.e., 2N × N includes the shares of both 2N × N and N × 2N CUs. The results are averaged over the four QP values.

Fig. 4 (a)–(c) depict the average shares of each CU size for intra and inter predicted CUs as a function of resolution, sequence, and QP value, respectively.

First off, Fig. 4 (a)–(c) show that CUs are predominantly encoded in inter. Moreover, intra CUs are on average smaller than inter CUs. Fig. 4 (a) confirms that video with higher resolutions use more large CUs. Indeed, high resolution leads to larger homogenous areas in spatial and temporal domain. In general, the resolution does not significantly affect the number of non-square CUs, but their average size increases with resolution. On average, 32 × 16 is the most used CU size with the largest three resolutions, 16 × 8 with the next two, and 8 × 4 with the smallest. Considering the decrease in resolution, the most popular CU size covers roughly the same area in a frame It is interesting to note that in most cases both 2N × N and 2N × N/2 are used more than their 2N × 2N counterparts, specifically with larger resolutions. This clearly

TABLE IV
TOOLS, THEIR ABBREVIATIONS, DESCRIPTIONS AND STUDY INCLUSIONS

Encoding Stages	Tool	Acronym or abbreviation	Short description	CU usage[1]	RDC[2]
Inter prediction	Uni-prediction	UniPred	UniPred uses one reference frame among past and future ones to predict a block.	×	
	Bi-prediction	BiPred	BiPred uses one or two reference frames among past and future ones to predict a block by blending to predictions.	×	
	Geometric partitioning mode	GPM	GPM allows to split the CU into two parts with an angled line.	×	×
	Affine motion	AM	AM allows predicting a CU with 4-parameter or 6-parameter affine transformation for non-translational motion.	×	×
	Merge with motion vector difference	MMVD	MMVD is a hybrid of merge and explicit signaling modes.	×	×
	Symmetric motion vector difference	SMVD	SMVD is used in the bidirectional prediction to save bits when the nearest references in both of the reference lists are used and the MVDs are point symmetric between the two references.	×	×
	Decoder-side motion vector refinement	DMVR	DMVR is used to improve the accuracy of the MVs of the merge mode of bidirectional prediction.		×
	Bi-directional optical flow	BDOF	BDOF is used to refine the bidirectional prediction signal of CU at the 4×4 subblock level.		×
Inter/Intra pred.	Combined Intra/Inter prediction	CIIP	CIIP combines an inter prediction signal with an intra prediction signal.	×	×
Intra prediction	Intra prediction	IntraPred	IntraPred includes 67 intra prediction modes: 65 angular, a planar, and a DC mode.	×	
	Cross-component linear model	CCLM	CCLM uses luma samples to predict chroma samples of the same CU in order to decrease cross-component redundancy.		×
	Matrix-based intra-picture prediction	MIP	MIP uses three steps to generate the prediction: averaging of reference lines, matrix vector multiplication, and interpolation.	×	
	Multiple reference lines	MRL	MRL allows intra prediction to use the second and fourth left and top lines of reference samples in addition to the traditional first line.	×	
	Intra sub-partitions	ISP	ISP allows luma TBs of intra predicted CUs to be split into two or four subblocks, depending on the size of the CU.	×	
Forward/inverse transform and quantization	Transform skip residual coding	TSRC	TSRC adapts the CABAC entropy coding of the special transform skip residual block to screen-content-specific characteristics.	×	×
	Multiple transform selection	MTS	MTS allows usage of the DST-VII and the DCT-VIII in addition to the default DCT-II.	×	×
	Non-separable secondary transform	LFNST	LFNST is an extra non-separable transform for the lowest frequencies.	×	×
	Subblock transform	SBT	SBT split the CU into two residual subblocks in vertical or horizontal direction in relation 2:2, 3:1, or 1:3. The smaller subblock forms a TU and the larger subblock is zeroed out.		×
	Joint coding of chroma residuals	JCCR	JCCR is used to reduce the redundancy of two similar chroma components' residual signals.	×	×
	Dependent quantization	DQ	DQ takes an advantage of the information of previous TUs to pack the transform coefficients more densely.		×
Loop filtering	Adaptive loop filter	ALF	ALF uses Wiener-based adaptive filter to minimize the mean squared error between the original and reconstructed samples.		×
	Luma mapping with chroma scaling	LMCS	LMCS is based on two steps: 1) in-loop mapping of the luma component and 2) luma-dependent chroma residual scaling.		×

[1] Tools included in the CU usage study in Section VI.
[2] Tools included in the RDC study in Section VII.

confirms the powerfulness of MTT in VVC, as otherwise the 2N × N CUs would require a QT split and coding the same information twice or 2N × N/2 would require two recursive QT splits and coding duplicate info for four CUs.

Resolution affects how the encoder reaches the final CTU structure, e.g., on average, about 20% and 25% of the 16 × 8 CUs are selected after three QT splits and a single MTT split with the 480p and 720p resolutions, whereas for 2160p it is less than 10%. Based on this, the QT+MTT traversal algorithms should consider the resolution of the sequence. Overall, resolution does not have a huge impact on whether intra or inter is used. The main difference is the CU size which is used roughly equally by smaller and larger resolutions. Since inter CUs are larger this happens at 32 × 8 and for intra at 32 × 4 and 16 × 8.

Since *Bosphorus* has the smallest temporal and spatial complexity, it is expected to have the greatest number of large CUs. Indeed, *Bosphorus* has the highest share of all CUs with the larger dimension being 128 or 64. Conversely, *DaylightRoad2* has the largest complexity thus it uses the most of smallest CUs. Because of the low spatial and high temporal coding complexity, *SunBath* uses the most intra CUs which by default reduces the usage of the largest CUs. 8 × 4 for *DaylightRoad2* and *SunBath* is the only case, where intra is used more than inter for a single CU size.

TABLE V
BREAKDOWN OF CU SIZES WITH DIFFERENT RESOLUTIONS IN VVENC ENCODER [11]

Seq.	Res.	128×128 2N×2N	128×64 2N×N N×2N	64×64 2N×2N	64×32 2N×N N×2N	64×16 2N×N/2 N/2×2N	64×8 2N×N/4 N/4×2N	64×4 2N×N/8 N/8×2N	32×32 2N×2N	32×16 2N×N N×2N	32×8 2N×N/2 N/2×2N	32×4 2N×N/4 N/4×2N	16×16 2N×2N	16×8 2N×N N×2N	16×4 2N×N/2 N/2×2N	8×8 2N×2N	8×4 2N×N N×2N	4×4 2N×2N
Bosphorus [35]	2160p	6.1%	1.4%	7.6%	11.9%	11.4%	4.8%	1.0%	8.6%	17.5%	9.4%	1.3%	7.8%	7.9%	1.4%	1.5%	0.3%	0.0%
	1440p	5.6%	1.1%	5.0%	7.9%	7.3%	5.2%	1.7%	7.0%	14.9%	11.3%	2.7%	8.5%	13.4%	3.5%	3.7%	1.1%	0.1%
	1080p	4.7%	1.0%	3.6%	6.3%	7.3%	6.6%	2.0%	5.9%	11.8%	10.7%	3.7%	8.0%	15.0%	5.4%	5.2%	2.5%	0.3%
	720p	3.7%	0.7%	5.5%	3.3%	6.2%	3.9%	2.1%	5.5%	9.0%	9.7%	5.1%	6.8%	15.6%	8.3%	7.4%	6.1%	1.2%
	480p	2.8%	0.5%	5.8%	5.1%	3.1%	3.2%	2.1%	4.6%	6.7%	9.0%	7.3%	5.7%	14.0%	10.6%	7.9%	9.3%	2.4%
	240p	1.9%	0.5%	4.1%	7.4%	6.1%	2.4%	2.2%	4.5%	6.4%	8.2%	8.9%	4.2%	11.5%	11.2%	7.2%	10.3%	2.9%
	Average	**4.2%**	**0.9%**	**5.3%**	**7.0%**	**6.9%**	**4.4%**	**1.8%**	**6.0%**	**11.0%**	**9.7%**	**4.8%**	**6.8%**	**12.9%**	**6.7%**	**5.5%**	**5.0%**	**1.1%**
DaylightRoad2 [6]	2160p	2.2%	1.8%	5.3%	6.9%	6.1%	3.1%	1.0%	5.7%	11.1%	8.8%	3.3%	7.5%	16.1%	7.6%	6.9%	5.7%	0.9%
	1440p	1.8%	1.5%	4.6%	6.6%	4.9%	2.8%	0.9%	5.9%	10.3%	8.7%	3.2%	7.4%	15.7%	7.9%	7.6%	8.3%	1.9%
	1080p	1.0%	1.0%	3.5%	5.8%	5.4%	3.6%	0.7%	5.6%	9.7%	8.4%	3.1%	7.2%	15.8%	8.6%	8.4%	9.9%	2.2%
	720p	0.5%	0.6%	3.5%	3.8%	4.8%	2.1%	0.5%	5.3%	9.0%	8.1%	3.2%	7.3%	16.0%	9.8%	9.5%	13.0%	2.9%
	480p	0.3%	0.3%	2.5%	4.4%	3.1%	1.9%	0.4%	4.8%	8.3%	7.4%	2.9%	7.2%	16.2%	10.4%	10.7%	15.7%	3.5%
	240p	0.1%	0.1%	1.5%	2.9%	3.9%	1.6%	0.3%	3.8%	7.7%	6.8%	2.7%	6.8%	15.3%	11.0%	11.8%	19.5%	4.4%
	Average	**1.0%**	**0.9%**	**3.5%**	**5.1%**	**4.7%**	**2.5%**	**0.6%**	**5.2%**	**9.4%**	**8.0%**	**3.1%**	**7.2%**	**15.9%**	**9.2%**	**9.1%**	**12.0%**	**2.6%**
ShakeNDry [35]	2160p	1.8%	0.8%	7.3%	10.5%	7.3%	1.9%	0.2%	13.3%	23.2%	8.3%	0.7%	11.6%	9.1%	1.5%	2.0%	0.6%	0.0%
	1440p	1.6%	0.4%	4.1%	6.2%	4.3%	1.9%	0.4%	10.4%	20.0%	11.1%	1.9%	12.8%	15.7%	3.9%	4.2%	1.0%	0.1%
	1080p	1.3%	0.3%	2.6%	4.0%	3.6%	2.3%	0.4%	7.8%	15.5%	10.8%	3.0%	11.8%	20.4%	7.1%	6.9%	2.3%	0.1%
	720p	1.0%	0.1%	2.1%	1.6%	2.3%	0.6%	0.2%	6.2%	11.1%	8.3%	3.5%	10.8%	21.8%	11.8%	10.9%	7.0%	0.7%
	480p	0.8%	0.1%	2.1%	2.0%	0.9%	0.4%	0.1%	4.9%	8.1%	6.3%	3.0%	9.9%	20.5%	13.6%	12.6%	13.6%	1.0%
	240p	0.4%	0.1%	2.4%	2.8%	2.2%	0.3%	0.0%	4.0%	6.9%	5.0%	2.5%	8.5%	20.7%	14.3%	14.3%	3.9%	1.6%
	Average	**1.1%**	**0.3%**	**3.4%**	**4.5%**	**3.4%**	**1.2%**	**0.2%**	**7.7%**	**14.1%**	**8.3%**	**2.4%**	**10.9%**	**18.0%**	**8.7%**	**8.5%**	**6.4%**	**0.6%**
SunBath [35]	2160p	1.4%	1.6%	7.8%	13.2%	8.1%	1.8%	0.2%	14.0%	21.9%	7.5%	0.7%	10.5%	7.9%	1.3%	1.7%	0.5%	0.0%
	1440p	0.6%	0.7%	3.9%	8.1%	5.5%	1.7%	0.3%	11.3%	21.3%	9.6%	1.3%	13.2%	13.8%	3.0%	4.1%	1.6%	0.2%
	1080p	0.3%	0.3%	2.4%	5.1%	4.3%	2.0%	0.3%	8.6%	18.5%	10.1%	1.6%	13.8%	17.7%	4.6%	6.5%	3.4%	0.5%
	720p	0.1%	0.1%	1.3%	2.4%	2.5%	0.9%	0.2%	6.0%	14.0%	8.8%	2.0%	13.1%	21.4%	7.4%	10.4%	8.0%	1.5%
	480p	0.0%	0.0%	0.6%	1.2%	1.1%	0.5%	0.1%	4.0%	9.2%	6.9%	2.0%	10.8%	22.0%	9.7%	13.9%	14.6%	3.2%
	240p	0%	0.0%	0.1%	0.4%	0.5%	0.2%	0.0%	1.9%	4.9%	3.9%	1.5%	7.7%	18.7%	11.4%	16.9%	25.8%	6.1%
	Average	**0.4%**	**0.5%**	**2.7%**	**5.1%**	**3.7%**	**1.2%**	**0.2%**	**7.6%**	**14.9%**	**7.8%**	**1.5%**	**11.5%**	**16.9%**	**6.3%**	**8.9%**	**9.0%**	**1.9%**
Tango [6]	2160p	3.4%	2.2%	8.9%	11.4%	8.5%	2.7%	0.5%	11.4%	19.0%	8.3%	0.9%	9.9%	9.1%	1.5%	1.9%	0.6%	0.0%
	1440p	2.1%	1.4%	6.3%	8.6%	5.6%	2.9%	0.4%	10.0%	17.0%	9.8%	1.4%	11.2%	14.4%	3.0%	4.1%	1.5%	0.2%
	1080p	1.2%	0.9%	4.8%	6.2%	5.4%	3.1%	0.4%	8.9%	15.0%	9.8%	1.8%	11.4%	17.3%	4.5%	6.1%	3.0%	0.4%
	720p	0.5%	0.4%	4.0%	3.3%	4.4%	1.9%	0.2%	7.6%	12.4%	8.8%	1.9%	11.2%	19.6%	6.6%	9.2%	6.7%	1.3%
	480p	0.2%	0.2%	2.4%	2.8%	2.3%	1.1%	0.1%	6.4%	9.9%	7.8%	1.9%	10.2%	20.2%	8.4%	12.1%	11.4%	2.5%
	240p	0.0%	0.0%	1.1%	1.4%	1.7%	1.0%	0.0%	3.7%	6.5%	6.0%	1.4%	8.4%	19.3%	9.8%	15.1%	19.3%	5.2%
	Average	**1.3%**	**0.8%**	**4.6%**	**5.6%**	**4.7%**	**2.1%**	**0.3%**	**8.0%**	**13.3%**	**8.4%**	**1.5%**	**10.4%**	**16.6%**	**5.6%**	**8.1%**	**7.1%**	**1.6%**
Average across sequences	2160p	3.0%	1.6%	7.4%	10.8%	8.3%	2.9%	0.6%	10.6%	18.5%	8.5%	1.4%	9.5%	10.0%	2.7%	2.8%	1.5%	0.2%
	1440p	2.4%	1.0%	4.8%	7.5%	5.5%	2.9%	0.7%	8.9%	16.7%	10.1%	2.1%	10.6%	14.6%	4.3%	4.8%	2.7%	0.5%
	1080p	1.7%	0.7%	3.4%	5.5%	5.2%	3.5%	0.7%	7.3%	14.1%	9.9%	2.6%	10.4%	17.2%	6.1%	6.6%	4.2%	0.7%
	720p	1.2%	0.4%	3.3%	2.9%	4.0%	1.9%	0.6%	6.1%	11.1%	8.8%	3.1%	9.8%	18.9%	8.8%	9.5%	8.2%	1.5%
	480p	0.8%	0.2%	2.7%	3.1%	2.1%	1.4%	0.6%	4.9%	8.4%	7.5%	3.4%	8.8%	18.6%	10.5%	11.5%	12.9%	2.5%
	240p	0.5%	0.1%	1.8%	3.0%	2.9%	1.1%	0.5%	3.6%	6.5%	6.0%	3.4%	7.1%	17.1%	11.6%	13.1%	17.8%	4.0%
	Overall average	**1.6%**	**0.7%**	**3.9%**	**5.4%**	**4.7%**	**2.3%**	**0.6%**	**6.9%**	**12.5%**	**8.4%**	**2.7%**	**9.4%**	**16.1%**	**7.3%**	**8.0%**	**7.9%**	**1.6%**

Like the resolution, increasing QP values increases the usage of larger CUs. Indeed, with higher QP values, each bit costs relatively more in the RD optimization. Overall, the impact of the QP value on the share of CU size mirrors that of the resolution. However, there are two major differences. First, QP value has a slightly higher effect on the QT+MTT depth. Secondly, unlike resolution, QP value has little effect on the QT+MTT traversal.

IV. VVC TOOL USAGE PER CU

Table VI tabulates the average usage of tools for each CU size. The results are averaged across all sequences, resolutions, and QP values. The cells with dash (-) indicate that the tool is forbidden for the specific CU size by the standard. Most of the tools follow a trend where the tool is selected more as the size of the CU increases or decreases. Additionally, MMVD, CIIP, JCCR, and the intra tools are also heavily based on the squareness of the CU. Fig. 5 (a)–(c) depict the average share of each tool as a function of resolution, sequence, and QP value, respectively. Any value under 1.5% is numbered in the figures.

BiPred, UniPred, and IntraPred share a relationship where UniPred and IntraPred are preferred with smaller CUs whereas BiPred with larger CUs. Since in the RA condition, the encoded frame is temporally between the reference frames, any regular movement is overwhelmingly better predicted with BiPred, i.e., any regular movement is mostly covered with a large CU using BiPred. On the other hand, when there is irregular movement, BiPred is unlikely to produce good results and smaller UniPred or intra predicted CUs are used. Increasing QP value reduces the amount of BiPred, because higher QP value reduces the quality of the reference. Thus, it is harder to find a blend between the two references that would offset the cost of coding two motion vectors. Enlarging the resolution increases the usage of BiPred because there are more details that can be preserved with BiPred.

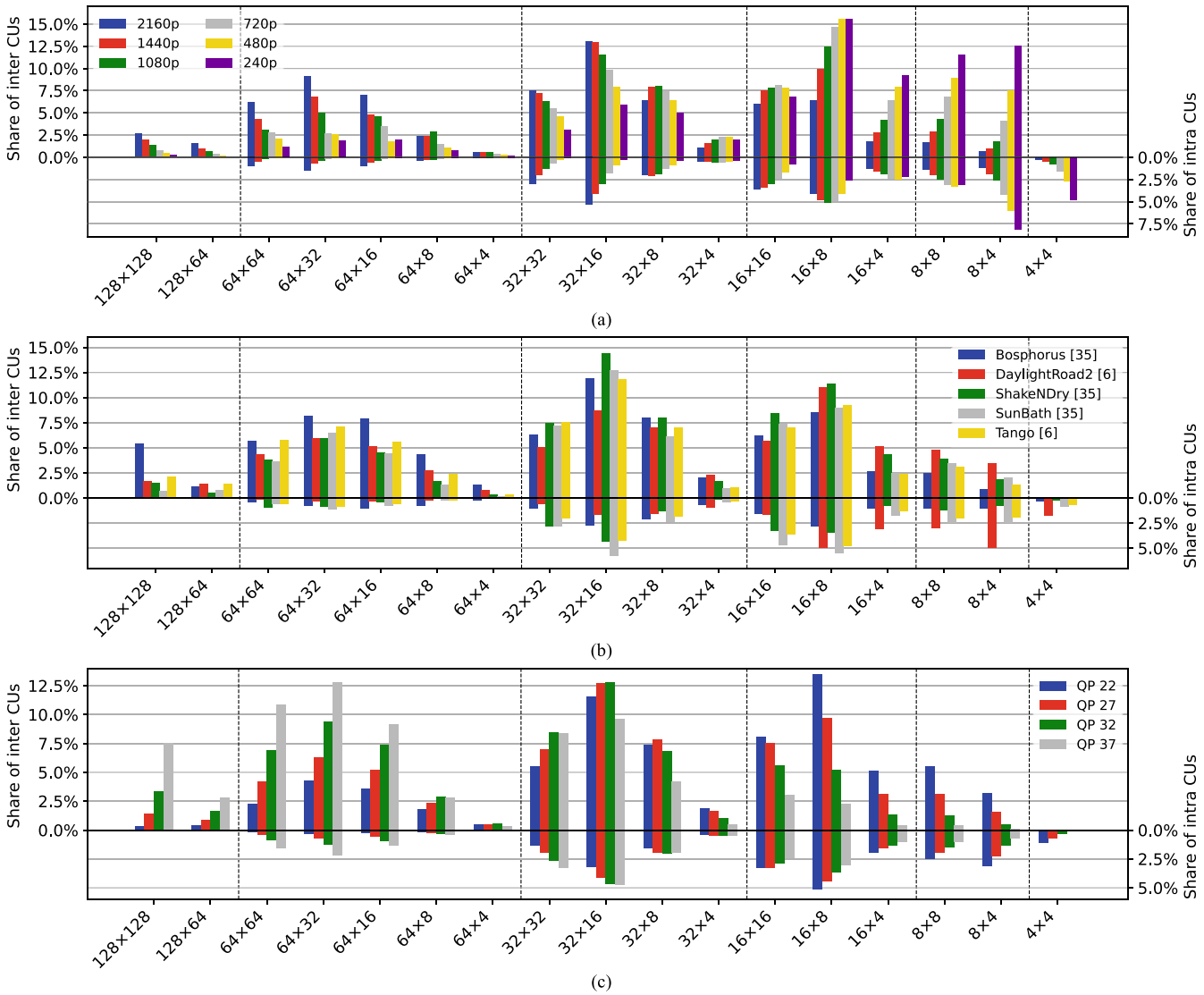


Fig. 4. Share of different sized inter and intra CUs of VVenC encoder [11]. (a) Per resolution. (b) Per sequence. (c) Per QP value.

TABLE VI
PERCENTAGE OF USED CODING TOOLS PER CU SIZE IN VVENC ENCODER [11]

	Inter prediction						Inter/intra	Intra prediction				Transform and residual			
CU	**UniPred**	**BiPred**	**GPM**	**AM**	**MMVD**	**SMVD**	**CIIP**	**IntraPred**	**MIP**	**MRL**	**ISP**	**TSRC**	**MTS**	**LFNST**	**JCCR**
128×128	14.34%	85.66%	-	22.30%	3.28%	2.21%	-	-	-	-	-	-	-	-	0.04%
128×64	19.52%	80.48%	-	20.24%	9.37%	5.68%	-	-	-	-	-	-	-	-	0.19%
64×64	17.97%	69.73%	1.93%	16.54%	7.04%	3.64%	3.84%	14.22%	5.79%	0.07%	0.48%	-	-	2.13%	1.97%
64×32	22.67%	63.55%	3.42%	16.17%	9.48%	4.41%	3.48%	13.84%	4.77%	0.11%	0.81%	-	-	3.44%	1.44%
64×16	25.87%	60.43%	2.90%	16.23%	8.89%	2.61%	2.78%	13.58%	2.61%	0.18%	1.83%	-	-	3.39%	1.18%
64×8	25.56%	62.12%	-	13.26%	9.51%	2.77%	3.66%	15.98%	2.18%	0.34%	2.09%	-	-	2.81%	1.45%
64×4	32.27%	54.92%	-	-	12.52%	3.05%	7.25%	20.06%	-	0.70%	2.23%	-	-	2.65%	1.30%
32×32	23.74%	46.04%	3.34%	11.99%	8.06%	2.81%	2.95%	29.83%	14.36%	0.25%	0.75%	0.22%	14.16%	5.79%	2.79%
32×16	27.00%	39.13%	4.87%	10.98%	8.92%	2.44%	3.15%	32.14%	11.75%	0.47%	1.75%	0.26%	12.81%	6.83%	2.02%
32×8	29.51%	40.86%	4.87%	8.80%	8.55%	1.75%	2.76%	27.52%	4.31%	0.68%	3.05%	0.42%	9.99%	4.46%	1.25%
32×4	34.90%	36.37%	-	-	11.06%	1.97%	5.50%	34.23%	3.85%	1.36%	4.64%	1.09%	9.50%	4.09%	1.00%
16×16	28.05%	31.21%	4.64%	9.58%	8.18%	1.45%	3.11%	39.20%	13.49%	0.72%	1.35%	0.38%	13.38%	10.35%	1.84%
16×8	29.21%	29.20%	6.19%	6.55%	8.82%	1.47%	3.03%	38.44%	9.08%	1.13%	2.45%	0.87%	14.63%	6.89%	1.31%
16×4	35.92%	27.51%	-	-	11.49%	1.50%	6.26%	42.83%	4.94%	1.46%	5.32%	1.81%	12.82%	5.52%	0.71%
8×8	32.17%	20.44%	6.01%	5.89%	9.97%	1.18%	3.14%	44.52%	8.08%	1.41%	2.17%	1.63%	16.68%	7.26%	1.06%
8×4	40.07%	-	-	-	12.13%	-	-	59.93%	10.56%	2.02%	9.50%	2.79%	18.95%	11.27%	0.19%
4×4	-	-	-	-	-	-	-	100.00%	12.73%	2.89%	-	3.46%	28.03%	20.84%	0.03%

Whereas most tools behave similarly between the different sequences, UniPred behaves differently between sequences, as depicted in Fig. 6. For *DaylightRoad2*, *ShakeNDry*, and *Tango* the encoder prefers non-square and smaller CUs, *Bosphorus* uses the least medium sized CUs, and *SunBath* uses UniPred fairly uniformly across all CU sizes. The behaviour of *DaylightRoad2*,

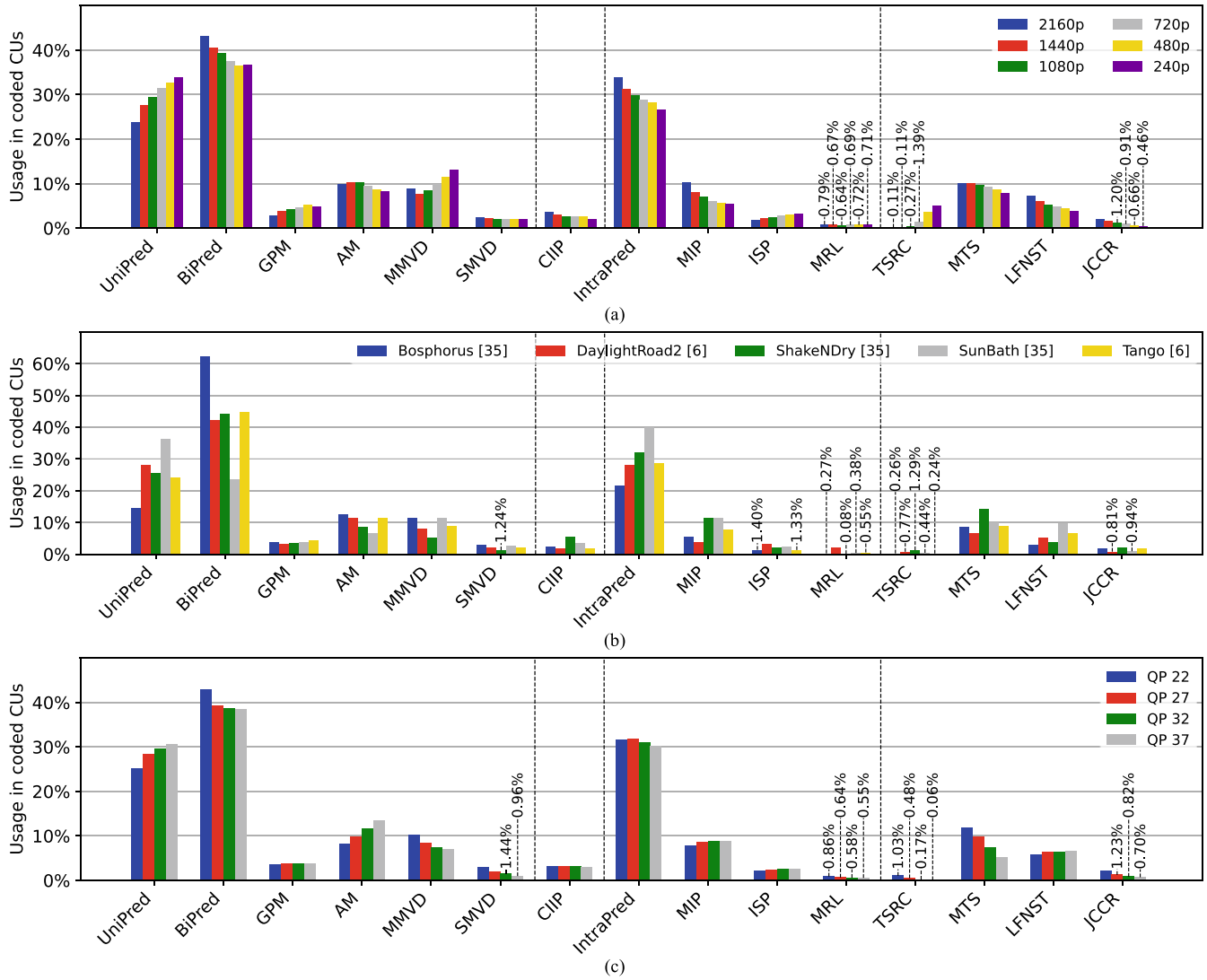


Fig. 5. Usage of coding tools of VVenC encoder [11]. (a) Per resolution. (b) Per sequence. (c) Per QP value.

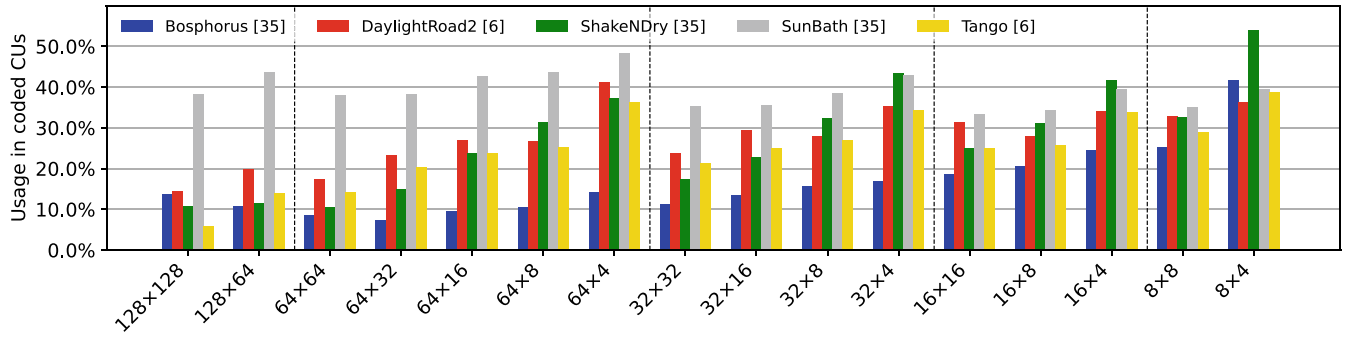
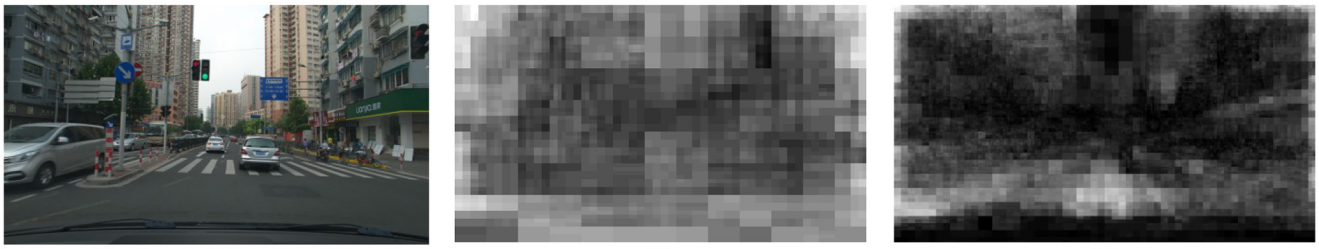


Fig. 6. Usage of UniPred for each sequence and CU size of VVenC encoder [11].

ShakeNDry, and *Tango* is as expected based on their coding complexities and the overall usage of UniPred. For *Bosphorus,* majority of the larger UniPred CUs are located at the left edge of the sequence, which is panned out of view, thus part of the future prediction would be missing. On the other hand, the smaller inter CUs are located near the shore, edges of the boat, and the bridge, which have a lot of detail. On *SunBath*, the UniPred CUs are fairly evenly distributed across the frame, with the exception that the largest CUs cover the bright area in the top right corner of the sequence. The large number of UniPred CUs is most likely because of the large difference in brightness between the branches moving on the foreground and the background. Additionally, the movement is quite large and erratic, which makes finding good BiPred difficult.



(a) (b) (c)

Fig. 7. First frame of *DaylightRoad2* and associated heatmap of affine CUs. (a) 2160p raw frame. (b) 240p heatmap. (c) 2160p heatmap.

The following tools are used more with less square and smaller CU size: 1) GPM, 2) MMVD, 3) CIIP, and 4) ISP. They are all used in scenarios where the HEVC inherited tools struggle to achieve a good quality. Such areas are likely to be irregularly shaped, e.g., edges of objects, thus the optimal CTU structure includes non-square CUs in those areas. 1) GPM is used more with smaller resolutions which indicates MTT splits are able to produce similar encoding efficiency with higher resolutions. Conversely, QP value does not affect the usage of GPM, indicating that reference quality does not affect the usage by much. The usage of GPM is directly linked to the number of defined edges in a sequence, which *Bosphorus* has the most: the boat, coastline, and the bridge in the background. On the other hand, *SunBath* has the least defined edges, since the large contrast difference softness the edges of the leaves enough that GPM is not able to work properly. 2) MMVD is used more with smaller resolutions, which correlates with the fact that it is used more with smaller CUs. MMVD is mostly used in scenarios where slight deviation to the merge candidate movement vector greatly improves the prediction quality, e.g., slightly irregular movement. Such detail is more likely preserved with lower QP value. On *Bosphorus,* it is mostly used on the water and on *SunBath* across the frame with slight bias to the branches. 3) CIIP is used slightly more with higher resolutions as there are more fine details to preserve. However, QP value does not affect its usage, again, indicating the reference quality does not affect the usability of the tool. *ShakeNDry* uses the most of CIIP due to the watersplashes, implying the tool is best used at chaotic areas that still have some uniformity in the texture. Finally, 4) ISP is used least of these four tools. As for GPM, it is used more with higher resolutions and not affected by QP value. In general, it is used similarly to GPM where the CU is split into two or four separate areas. On *DaylightRoad2*, most of ISP is used on the buildings on the edges, the area between the onboard car, and the road.

AM is used most with large CUs, particularly on *DaylightRoad2*, which features scaling in particularly at the edges of the frame as the camera moves forward. AM is mostly used on *Bosphorus* at the bottom of the frame where the wave movement towards the camera is most pronounced, and on *Tango* on the non-translational movement of the dancers. Even though *ShakeNDry* and *SunBath* neither have obvious non-translational movement, AM is still used in those sequences indicating that AM could be sometimes beneficial for predicting translational movement. AM is used relatively more with lower resolutions for two reasons: the AM CUs are on average smaller compared with the average size of the CUs and it can be used on larger areas as shown in Fig. 7.

There are only couple of note-worthy points about SMVD: 1) lower QP value increases its usage more than what would be implied by the increase in general usage of BiPred, and 2) *ShakeNDry* uses the least SMVD because it has the least regular movement.

Overall, MRL is not used much, especially with larger CUs. The usage correlates to some extent with AM, except when resolution is considered, and it is used in same areas as AM in *DaylightRoad2*.

In general, JCCR is used little, though it is used more with higher resolution, lower QP value, and square CUs. TSRC is mostly intended for screen content coding; thus, the selected dataset is not optimal for evaluating TSRC, but it will still indicate how good it is for general encoding. TSRC is used most with smaller resolutions and CUs. Considering that only 0.22% of 32 $\times$ 32 CUs use TSRC, it is understandable that the TSRC is limited to 32 $\times$ 32 CUs. MTS is more used with smaller CUs indicating that it is used more in scenarios that are difficult to predict. As for TSRC, it is used more with smaller QP value, i.e., increasing quantization reduces the importance of the transform type. *ShakeNDry* uses the most MTS because of the water splashes and higher resolutions use more MTS because there is more detail to preserve. Considering that 14.16% of 32 $\times$ 32 CUs still use MTS, it might have been beneficial for the standard to include MTS for the 64 $\times$ N CUs and maybe allow limiting the maximum size of the used transforms at the sequence level. In fact, they were originally included but removed quite early due to complexity concerns. However, since the complexity was increased a lot after that, it would have been worth it to revisit the 64 $\times$ 64 transforms.

LFNST is used more with higher resolution and QP value. Sequence wise, it is used most in *SunBath* that has the highest temporal coding complexity [35]. However, it is used the second most in *Tango*, indicating that the usability of LFNST is not related to temporal or spatial coding complexity.

Disabling a tool affect the size of the encoded CUs. In most cases the effect is fairly small and predictable, e.g., disabling a BiPred tool slightly reduces the number of BiPred CUs. However, with DQ the effect is most significant. Fig. 8 depicts the relative change in CU usage for different resolutions when DQ is disabled. There is a clear trend that disabling DQ

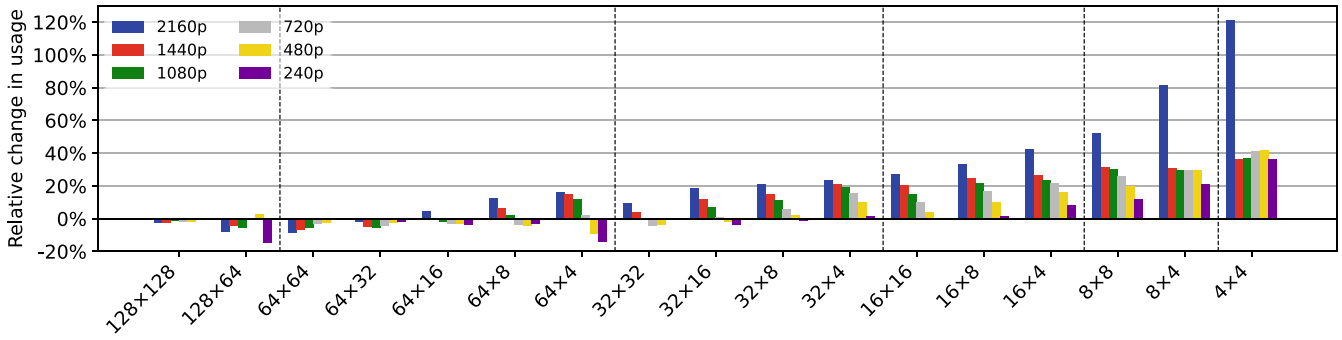


Fig. 8. Change in the CU usage breakdown when dependent quantization is disabled in VVenC encoder [11].

reduces the average size of CUs, so any algorithm predicting the CTU structure should take into account whether DQ is enabled or not.

V. RDC Analysis of VVC Tools

Table VII tabulates the average BD-rates and encoding speeds when each of the tools is disabled at a time. In addition, Table VII summarizes the overall average results across QP values, resolutions, and sequences. Positive BD-rate indicates how much the BD-rate worsens when the tool is disabled. Speed over one indicates the encoder speeds up when the tool is disabled. In most cases, disabling tool reduces the complexity and degrades BD-rate. However, disabling TSRC increases the complexity but also slightly improves the BD-rate in most cases, whereas disabling LMCS reduces complexity and improves quality.

On average, most tools reduce their complexity overhead as the resolution increases. These findings are in line with our previous study [4]; deeper QT+MTT search and relatively larger initialization overheads increase the complexity at low resolutions. The major exceptions are AM, CIIP, LFNST, LMCS, MMVD, and SMVD. On *DaylightRoad2* and *Tango,* AM complexity reduces as resolution grows but on the other sequences complexity increases. These sequences do not improve with AM indicating that the encoder has to perform a full AM search. The overall complexity of CIIP is too small to draw conclusions. LFNST has the smallest complexity at 720p indicating that the increased usage at higher resolutions causes significant overhead. Relative complexity of LMCS decreases as resolutions increase, except when increasing the resolution from 1440p to 2160p, overall, the complexity varies a lot between sequences and resolutions so predicting the complexity of LMCS is difficult. Neither MMVD nor SMVD complexities are affected by resolution in a meaningful way.

The tools can be separated into three categories depending on the effect of the resolution on the PSNR BD-rate: 1) bigger improvement with larger resolutions: ALF, AM, CIIP, LFNST, LMCS, MMVD, SMVD, and TR-skip; 2) independent of resolution: DQ, CCLM, MTS, and JCCR; and 3) smaller improvement with larger resolution: BDOF, DMVR, GPM, and SBT. The categories 1) and 2) are predictable, as our previous study [4] showed that reducing the resolution reduces BD-rate gain on average. However, category 3) is unexpected. BDOF and DMVR are limited to BiPred, and since it is used more with smaller resolutions, BDOF and DMVR can improve the quality more. GPM is also used more with smaller resolutions. SBT is also limited to inter predicted CUs, so it is used relatively more with smaller resolutions. Since the common factor with all of the category 3) tools is that they are used more with smaller resolutions, one would expect that TSRC, which is also used more with smaller resolutions, would belong to category 3). However, since TSRC is an alternative to MTS and MTS is less used with smaller resolutions there are more chances to benefit from TSRC.

AM and DMVR have the greatest relative variance in all BD-rates between the sequences, excluding LMCS; *DaylightRoad2* benefits the most of AM, *Tango* the second most, and rest of the sequences very little or not at all. Although all sequences use AM roughly the same amount, the benefit greatly varies between the sequences, but not the complexity. This indicates that light analysis could be performed to detect whether the sequence contains non-translational movement before the actual AM search is performed. Both *DaylightRoad2* and *Tango* benefit greatly from DMVR, but *ShakeNDry* hardly benefits at all, or decreases the quality in the case of VMAF BD-rate. Since DMVR is not signaled individually for each CU, the encoder could perform a careful pre-analysis of the sequence whether to turn it on. BDOF is similar, although the variance is slightly lower, so the same considerations should be applied to it. Finally, JCCR has much higher complexity on *DaylightRoad2* and *Tango*, especially on lower resolutions, even though it is not widely used and provides low BD-rate improvement. Thus, JCCR should probably be disabled when encoding sequences with smaller resolutions.

ShakeNDry has the largest average encoding time, thus it has the smallest complexity overhead for almost all tools. Indeed, the relative overhead of each tool is smaller, except for LMCS and JCCR. The difference of average encoding time between the rest of the sequences is small. Therefore, even though *Bosphorus* has the smallest average encoding time, it has largest overhead only with ALF. AM, JCCR, and DQ have the largest variance in complexity between sequences. DQ has the lowest complexity overhead on *ShakeNDry*, and highest on *DaylightRoad2*, contrarily to the MTS usage of the sequences. Since *ShakeNDry* and *Bosphorus* have the best BD-rate improvement with DQ, it is most beneficial with sequences with chaotic content that have high frequencies in the transform.

TABLE VII
RATE-DISTORTION-COMPLEXITY OF INDIVIDUAL CODING TOOLS OVER RESOLUTIONS AND SEQUENCES IN VVENC ENCODER [11]

				GPM	AM	MMVD	SMVD	DMVR	BDOF	CIIP	CCLM	TSRC	MTS	LFNST	SBT	JCCR	DQ	ALF	LMCS
Results averaged per resolution	2160p	BD-rate	PSNR	0.4%	1.9%	0.6%	0.2%	1.0%	0.6%	0.3%	2.1%	0.0%	0.7%	0.8%	0.4%	0.1%	2.2%	4.1%	0.0%
			SSIM	0.5%	1.9%	0.7%	0.2%	1.0%	0.6%	0.3%	3.9%	0.0%	0.7%	0.9%	0.3%	0.3%	2.2%	7.7%	-0.5%
			VMAF	0.3%	1.8%	0.5%	0.2%	0.9%	0.8%	0.4%	0.5%	0.0%	0.5%	0.7%	0.4%	0.1%	1.8%	4.4%	0.3%
			Speed	1.04×	1.19×	1.07×	1.04×	1.05×	1.07×	1.00×	1.05×	0.95×	1.14×	1.06×	1.08×	1.05×	1.08×	1.07×	1.16×
	1440p	BD-rate	PSNR	0.6%	2.2%	0.5%	0.2%	1.1%	0.7%	0.2%	2.1%	-0.1%	0.8%	0.6%	0.5%	0.1%	2.3%	3.6%	-0.6%
			SSIM	0.7%	2.2%	0.6%	0.2%	1.2%	0.8%	0.2%	4.1%	-0.2%	0.8%	0.7%	0.3%	0.3%	2.2%	6.1%	-0.1%
			VMAF	0.6%	2.2%	0.5%	0.2%	1.2%	1.0%	0.3%	0.6%	-0.1%	0.7%	0.7%	0.5%	0.2%	1.9%	4.4%	-1.1%
			Speed	1.06×	1.20×	1.08×	1.05×	1.08×	1.10×	1.01×	1.08×	0.94×	1.17×	1.05×	1.08×	1.05×	1.11×	1.10×	1.11×
	1080p	BD-rate	PSNR	0.8%	2.2%	0.4%	0.2%	1.2%	0.8%	0.2%	2.2%	-0.1%	1.0%	0.5%	0.6%	0.2%	2.3%	3.5%	-0.7%
			SSIM	0.9%	2.1%	0.5%	0.3%	1.4%	0.9%	0.2%	4.5%	-0.1%	0.8%	0.7%	0.3%	0.3%	2.3%	5.4%	-0.3%
			VMAF	0.9%	2.1%	0.4%	0.3%	1.4%	1.3%	0.2%	0.7%	0.1%	0.6%	0.7%	0.5%	0.2%	1.9%	4.1%	-0.6%
			Speed	1.06×	1.20×	1.08×	1.04×	1.08×	1.11×	1.00×	1.09×	0.94×	1.17×	1.05×	1.09×	1.05×	1.14×	1.12×	1.15×
	720p	BD-rate	PSNR	0.8%	2.0%	0.3%	0.2%	1.2%	1.0%	0.1%	2.0%	-0.1%	0.8%	0.4%	0.5%	0.1%	2.3%	3.1%	-0.8%
			SSIM	1.1%	2.0%	0.4%	0.1%	1.5%	1.1%	0.1%	4.6%	-0.1%	0.7%	0.6%	0.1%	0.4%	2.2%	4.3%	-0.4%
			VMAF	1.0%	1.8%	0.3%	0.2%	1.6%	1.8%	0.3%	0.7%	0.0%	0.4%	0.6%	0.3%	0.0%	2.0%	3.6%	-1.1%
			Speed	1.06×	1.19×	1.06×	1.03×	1.11×	1.12×	1.01×	1.10×	0.93×	1.21×	1.04×	1.10×	1.06×	1.21×	1.15×	1.15×
	480p	BD-rate	PSNR	0.9%	2.0%	0.4%	0.2%	1.3%	1.1%	0.1%	2.1%	-0.1%	0.8%	0.5%	0.5%	0.2%	2.3%	2.7%	-0.9%
			SSIM	1.3%	2.0%	0.4%	0.2%	1.6%	1.3%	0.0%	5.1%	-0.1%	0.7%	0.7%	0.1%	0.6%	2.1%	3.2%	-0.3%
			VMAF	1.1%	1.8%	0.5%	0.2%	1.3%	2.4%	-0.1%	0.7%	0.2%	0.7%	0.7%	0.5%	0.2%	2.1%	3.5%	-1.0%
			Speed	1.06×	1.19×	1.07×	1.04×	1.13×	1.15×	1.01×	1.14×	0.93×	1.26×	1.05×	1.10×	1.08×	1.27×	1.19×	1.19×
	240p	BD-rate	PSNR	0.9%	1.9%	0.4%	0.1%	1.4%	1.4%	0.1%	2.1%	-0.2%	0.8%	0.3%	0.7%	0.2%	2.2%	1.8%	-0.8%
			SSIM	1.2%	1.9%	0.4%	0.1%	2.0%	1.5%	0.1%	6.0%	-0.3%	0.6%	0.4%	0.3%	0.3%	2.1%	1.8%	-0.4%
			VMAF	1.3%	1.7%	0.4%	0.2%	1.6%	2.6%	-0.1%	0.4%	-0.1%	0.5%	0.0%	0.6%	0.6%	1.6%	3.5%	-1.3%
			Speed	1.07×	1.18×	1.07×	1.04×	1.18×	1.20×	1.01×	1.18×	0.92×	1.32×	1.07×	1.16×	1.09×	1.36×	1.26×	1.21×
Results averaged per sequences	*Bosphorus* [35]	BD-rate	PSNR	0.3%	0.5%	0.4%	0.1%	0.8%	0.6%	0.1%	3.6%	-0.1%	0.9%	0.2%	0.4%	0.3%	2.8%	1.9%	0.1%
			SSIM	0.3%	0.3%	0.6%	0.1%	0.9%	0.8%	0.1%	7.5%	0.0%	0.9%	0.2%	0.1%	0.5%	2.8%	2.6%	1.3%
			VMAF	0.9%	0.0%	0.5%	0.2%	0.9%	2.5%	0.1%	1.2%	0.3%	0.6%	0.6%	0.6%	0.5%	2.7%	5.1%	-1.2%
			Speed	1.06×	1.14×	1.07×	1.02×	1.12×	1.14×	1.01×	1.10×	0.92×	1.22×	1.05×	1.07×	1.01×	1.20×	1.17×	1.14×
	Daylight-Road2 [6]	BD-rate	PSNR	0.7%	7.0%	0.6%	0.2%	2.8%	2.0%	0.1%	0.6%	-0.2%	0.6%	0.6%	0.6%	0.3%	1.6%	4.2%	-0.2%
			SSIM	0.6%	7.2%	0.6%	0.1%	3.2%	2.1%	0.1%	1.4%	-0.3%	0.4%	0.7%	0.3%	0.8%	1.5%	6.4%	-0.3%
			VMAF	0.6%	7.2%	0.8%	0.3%	3.4%	2.9%	0.0%	0.0%	0.0%	0.6%	0.6%	0.4%	0.4%	1.5%	6.0%	-0.3%
			Speed	1.06×	1.27×	1.06×	1.03×	1.11×	1.12×	1.00×	1.11×	0.94×	1.23×	1.05×	1.13×	1.13×	1.26×	1.15×	1.17×
	ShakeNDry [35]	BD-rate	PSNR	0.5%	0.0%	0.0%	0.1%	0.1%	0.0%	0.2%	2.4%	-0.1%	1.0%	0.3%	0.7%	0.2%	3.8%	2.6%	-2.3%
			SSIM	0.6%	0.0%	0.0%	0.1%	0.1%	0.0%	0.1%	5.7%	-0.1%	0.8%	0.5%	0.4%	0.3%	4.0%	4.6%	-2.8%
			VMAF	0.5%	-0.2%	-0.3%	-0.1%	-0.2%	0.1%	0.2%	0.8%	-0.1%	0.6%	0.1%	0.3%	-0.1%	2.4%	1.9%	-0.7%
			Speed	1.04×	1.11×	1.07×	1.03×	1.07×	1.10×	1.00×	1.09×	0.91×	1.19×	1.04×	1.09×	1.01×	1.16×	1.12×	1.22×
	SunBath [35]	BD-rate	PSNR	0.7%	0.6%	0.6%	0.3%	0.3%	0.3%	0.3%	1.0%	-0.1%	0.8%	0.5%	0.5%	0.0%	1.6%	5.0%	-0.2%
			SSIM	1.1%	0.6%	0.7%	0.4%	0.5%	0.4%	0.2%	3.0%	-0.1%	0.6%	0.6%	0.4%	-0.1%	1.3%	8.0%	0.1%
			VMAF	0.9%	0.8%	0.7%	0.4%	0.5%	0.6%	0.3%	0.2%	0.0%	0.7%	0.5%	0.6%	0.2%	1.5%	3.1%	-0.5%
			Speed	1.09×	1.17×	1.09×	1.07×	1.13×	1.15×	1.03×	1.13×	0.97×	1.21×	1.07×	1.11×	1.02×	1.17×	1.16×	1.14×
	Tango [6]	BD-rate	PSNR	1.5%	2.0%	0.6%	0.2%	2.2%	1.7%	0.2%	2.8%	-0.2%	0.8%	1.0%	0.4%	0.1%	1.5%	1.9%	-0.5%
			SSIM	2.1%	2.0%	0.7%	0.2%	2.6%	1.8%	0.2%	5.8%	-0.1%	0.8%	1.2%	0.0%	0.3%	1.4%	2.1%	0.1%
			VMAF	1.5%	1.7%	0.5%	0.2%	2.2%	2.1%	0.1%	0.8%	0.0%	0.5%	1.1%	0.4%	0.1%	1.4%	3.6%	-1.4%
			Speed	1.05×	1.26×	1.07×	1.05×	1.09×	1.12×	1.00×	1.10×	0.93×	1.21×	1.05×	1.10×	1.14×	1.19×	1.15×	1.13×
Overall averaged results		**BD-rate**	**PSNR**	**0.7%**	**2.0%**	**0.4%**	**0.2%**	**1.2%**	**0.9%**	**0.2%**	**2.1%**	**-0.1%**	**0.8%**	**0.5%**	**0.5%**	**0.2%**	**2.3%**	**3.1%**	**-0.6%**
			SSIM	**0.9%**	**2.0%**	**0.5%**	**0.2%**	**1.5%**	**1.0%**	**0.2%**	**4.7%**	**-0.1%**	**0.7%**	**0.7%**	**0.2%**	**0.4%**	**2.2%**	**4.8%**	**-0.3%**
			VMAF	**0.9%**	**1.9%**	**0.4%**	**0.2%**	**1.3%**	**1.6%**	**0.2%**	**0.6%**	**0.0%**	**0.6%**	**0.6%**	**0.5%**	**0.2%**	**1.9%**	**3.9%**	**-0.8%**
			Speed	**1.06×**	**1.19×**	**1.07×**	**1.04×**	**1.10×**	**1.12×**	**1.01×**	**1.10×**	**0.93×**	**1.21×**	**1.05×**	**1.10×**	**1.06×**	**1.19×**	**1.15×**	**1.16×**

Fig. 9 depicts the average RDC performance of each tool in term of encoding speed and PSNR, SSIM, and VMAF BD-rates in blue, red, and green, respectively. Overall, CCLM, when not measured with VMAF, ALF, and DQ offer the highest improvement in BD-rate. However, ALF and DQ are also among the tools with highest complexities. LFNST, GPM, and DMVR are also able to provide relatively good RDC tradeoff. On the other hand, LMCS and MTS have the worst RDC tradeoff, thus their usage should be considered on a case-by-case basis.

In general, the PSNR, SSIM, and VMAF BD-rates are consistent, with the major exception of CCLM that improves the BD-rates by 2.1%, 4.7%, and 0.7%, respectively. Additionally, ALF improves SSIM, particularly with large resolutions, and VMAF BD-rates more than PSNR. Considering VVC overall improves the subjective quality more than objective [5], [41], one would expect that the VMAF BD-rate would be noticeably better than PSNR BD-rate, especially for the newly introduced tools. However, BDOF and ALF are the only tools that have noticeably better VMAF BD-rate. This indicates that there is still room for improving the RD performance to include more perceptual features. Additionally, *ShakeNDry* has on average the worst VMAF BD-rate, indicating that some of the tools may not be able to perform optimal rate-distortion

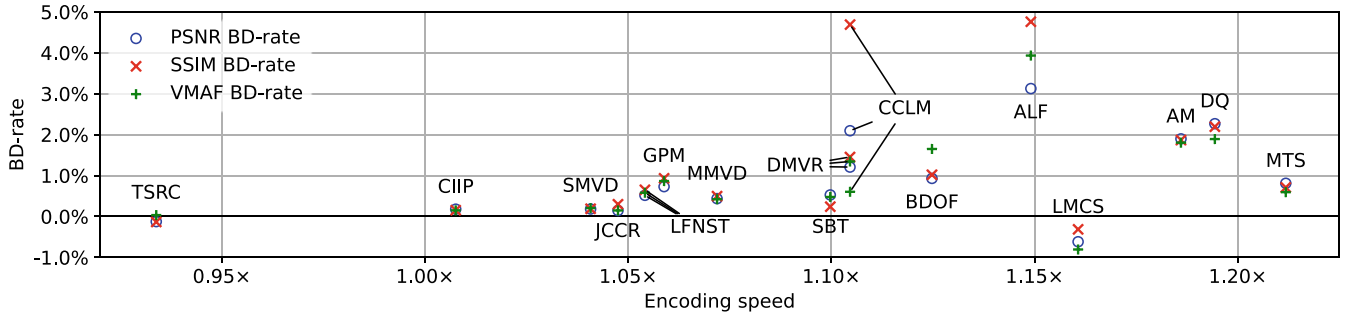


Fig. 9. RDC performance of the tools of VVenC encoder [11].

optimization for perceptual quality because of the chaotic water droplets.

Overall, considering the poor performance of LMCS, the parameter selection algorithm does not seem to be universally optimal and the characteristic of the different sequences in this study can act as a starting point for any further investigation to LMCS.

VI. Implementation Guidelines

Our analysis is intended to give guidelines on implementation priority and optimization potential of coding tools in a practical VVC encoder. The analysis outcome together with the fact that MTT provides the largest coding gain [8] indicate that MTT is a pivotal part of the encoder. MTT also affects nearly all other tools, so it is better to be implemented as soon as possible.

MTT also has the largest impact on the encoding complexity, and there are three viable approaches to mitigate it: 1) reduce the maximum search depth; 2) use temporal and spatial content features and already coded CTUs to predict the most likely MTT structures and only test them; and 3) limit the tools used for different CU sizes. The first approach is straightforward, but it only results in a suboptimal implementation. The other two approaches are more laborious, but also more profitable.

The RDC analysis in Section V can be used as a starting point for the second approach. Both resolution and QP value affect the average size of CUs linearly, so it is recommended to take them into account. Video content also has a large effect on the CTU structure, both locally and globally. The temporal and spatial coding complexity are global, e.g., *Bosphorus* has the lowest complexity that comes up with largest average CU sizes. On the other hand, the fact that small resolution versions of *Bosphorus* use the most 64 × 4 CUs is caused by the local features, i.e., the waves moving towards the camera form a single narrow continuous area. Therefore, the coding complexity can be used to predict the average size of the CUs, but other methods such as machine learning needs to be used to predict the local features that determine the CTU structure.

Similarly, the analysis in Section VI can be used as a starting point for the third approach. For example, our results showed that GPM, MMVD, CIIP, and ISP are used more with non-square CUs so they can be disabled for square CUs for a lesser RD penalty. Resolution and QP value affect the usage of most of the tools only slightly, so it is difficult to predict the tool usage based on these features.

The RDC performance is one of the key factors in choosing the tools for an encoder. As shown in Fig. 9, CCLM, ALF, and GPM offer the best RDC tradeoff and are thereby among the first tools to be implemented. On the other hand, the universality of the tool also affects the priority, e.g., AM is used fairly equally between all sequences, but it only provides excellent RDC tradeoff with *DaylightRoad2*. Therefore, AM should be enhanced with an algorithm that is able to identify non-translational movement. Conversely, LMCS, SBT, and MTS have the worst RDC tradeoffs so their implementations should be left last. Furthermore, the complexity of MTS should be reduced since only about 15% of CUs use it. For example, heuristically performing the MTS search for only 30% of the most potential CUs should bring the RDC tradeoff in line with other tools. In addition, the implementation effort should be considered when choosing a tool to be implemented, but it is left out of the scope of our analysis.

For optimization, the obvious choices are AM and DQ since they provide a good RD performance but with relatively high complexity. In general, there are more optimization opportunities for tools that perform non-normative operations, e.g., the optimization of BDOF is limited to the prediction generation, whereas both the prediction generation and the search can be optimized in AM. Amdahl's law is also an important factor when choosing which tools to optimize, e.g., optimizing CIIP first is hardly beneficial as it only accounts for 1% of the total encoding time.

As with HEVC, the energy consumption of VVC is higher than that of its predecessors. Corrêa *et al.* [33] were able to reduce the energy requirement of HEVC by 80% for minimal visual loss. Because VVC has significantly more tools and possible CTU structures, its energy saving potential is significantly higher. Therefore, the analysis presented in this paper is a pivotal starting point for implementing energy efficient software and hardware VVC encoders. First hardware VVC encoders are unlikely to include all available tools, so our analysis can be used to determine the implementation priority.

VII. Conclusion

This paper performed design space exploration for a practical VVenC encoder by focusing on its novel QT+MTT coding scheme and related coding tools in the VVC RA coding

condition. In practice, the DSE was carried out by analyzing the distribution of coding blocks, coding tool usage per block size, and the RDC characteristics of each tool as a function of versatile test set.

Our main conclusions are:

1) The implementation and optimization of a VVC encoder should be started from the MTT;
2) The complexity of the MTT can be reduced by predicting the block structure and used coding tools from the sequence features;
3) CCLM, ALF, and GPM are among the first coding tools to be implemented;
4) AM should be accompanied with an algorithm that detects non-translational movement; and
5) Coding tools that implement search functionality, e.g., AM and ALF are the most potential candidates for optimization.

The analysis in this paper can be used as a starting point for any work aiming to implement or reduce the complexity of VVC software and hardware encoders.

REFERENCES

[1] "Cisco visual networking index: Forecast and trends, 2017–2022," Cisco Syst., San Jose, CA, USA, White Paper, Dec. 2018. Accessed: Jan. 28, 2022. [Online]. Available: http://web.archive.org/web/20181213105003/https:/www.cisco.com/c/en/us/solutions/collateral/service-provider/visual-networking-index-vni/white-paper-c11-741490.pdf

[2] *Versatile Video Coding, ISO/IEC 23090-3 (VVC)*, Rec. H.266, Int. Telecommun. Union, Geneva, Switzerland, Jul. 2020.

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

[4] A. Mercat, A. Mäkinen, J. Sainio, A. Lemmetti, M. Viitanen, and J. Vanne, "Comparative rate-distortion-complexity analysis of VVC and HEVC video codecs," *IEEE Access*, vol. 9, pp. 67813–67828, 2021.

[5] N. Sidaty, W. Hamidouche, O. Déforges, P. Philippe, and J. Fournier, "Compression performance of the versatile video coding: HD and UHD visual quality monitoring," in *Proc. Picture Coding Symp.*, Nov. 2019, pp. 1–5.

[6] F. Bossen, J. Boyce, K. Suehring, X. Li, and V. Seregin, "VTM common test conditions and software reference configurations for SDR video," Int. Telecommun. Union, Geneva, Switzerland, document JVET-T2010, ITU-T SG16 WP3, ISO/IEC JTC1/SC29/WG11, Oct. 2020.

[7] Y.-W. Huang *et al.*, "Block partitioning structure in the VVC standard," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 31, no. 10, pp. 3818–3833, Oct. 2021.

[8] J. Brandenburg *et al.*, "Towards fast and efficient VVC encoding," in *Proc. IEEE Int. Workshop Multimedia Signal Process.*, Tampere, Finland, Sep. 2020, pp. 1–6.

[9] "20/20 Vision for Mobile Video [U.S.]" Snap. Accessed: Jan. 20, 2022. [Online]. Available: https://forbusiness.snapchat.com/blog/us-2020-vision-for-mobile-video

[10] "VVC Reference Software Version 10.0." [Online]. Available: https://vcgit.hhi.fraunhofer.de/jvet/VVCSoftware_VTM/-/tree/VTM-10.0 (Accessed: Jan. 20, 2022).

[11] A. Wieckowski *et al.*, "VVenC: An open and optimized VVC encoder implementation," in *Proc. IEEE Int. Conf. Multimedia Expo Workshops*, Shenzhen, China, Jul. 2021, pp. 1–2.

[12] J. Brandenburg, A. Wieckowski, T. Hinz, and B. Bross. "VVenC Fraunhofer Versatile Video Encoder." 2020. [Online]. Available: https://www.digitalmedia.fraunhofer.de/content/dam/dcinema/en/documents/ibc2020/VVenC_Versatile-Video-Encoder_Paper.pdf

[13] D. García-Lucas, G. Cebrián-Márquez, and P. Cuenca, "Rate-distortion/complexity analysis of HEVC, VVC and AV1 video codecs," *Multimedia Tools Appl.*, vol. 79, no. 39, pp. 29621–29638, Aug. 2020.

[14] T. Laude, Y. G. Adhisantoso, J. Voges, M. Munderloh, and J. Ostermann, "A comprehensive video codec comparison," *APSIPA Trans. Signal Inf. Process.*, vol. 8, p. e30, Nov. 2019.

[15] F. Bossen, X. Li, K. Sühring, K. Sharman, V. Seregin, and A. Tourapis, "JVET AHG report: Test model software development (AHG3)," Int. Telecommun. Union, Geneva, Switzerland, document JVET-U0003, ITU-T SG16 WP3, ISO/IEC JTC1/SC29/WG11, Jan. 2021.

[16] A. Cerveira, L. Agostini, B. Zatt, and F. Sampaio, "Memory assessment of versatile video coding," in *Proc. IEEE Int. Conf. Image Process.*, Abu Dhabi, UAE, Oct. 2020, pp. 1186–1190.

[17] F. Pakdaman, M. A. Adelimanesh, M. Gabbouj, and M. R. Hashemi, "Complexity analysis of next-generation VVC encoding and decoding," in *Proc. IEEE Int. Conf. Image Process.*, Abu Dhabi, UAE, Oct. 2020, pp. 3134–3138.

[18] Í. Siqueira, G. Correa, and M. Grellert, "Rate-distortion and complexity comparison of HEVC and VVC video encoders," in *Proc. Latin Amer. Symp. Circuits Syst.*, Feb. 2020, pp. 1–4.

[19] F. Bossen, K. Sühring, A. Wieckowski, and S. Liu, "VVC complexity and software implementation analysis," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 31, no. 10, pp. 3765–3778, Oct. 2021.

[20] M. Saldanha, G. Sanchez, C. Marcon, and L. Agostini, "Complexity analysis Of VVC intra coding," in *Proc. EEE Int. Conf. Image Process.*, Abu Dhabi, UAE, Oct.2020, pp. 3119–3123.

[21] A. Tissier, A. Mercat, T. Amestoy, W. Hamidouche, J. Vanne, and D. Menard, "Complexity reduction opportunities in the future VVC intra encoder," in *Proc. IEEE Int. Workshop Multimedia Signal Process.*, Sep. 2019, pp. 1–6.

[22] Z. Wang, A. C. Bovik, H. R. Sheikh, and E. P. Simoncelli, "Image quality assessment: From error visibility to structural similarity," *IEEE Trans. Image Process.*, vol. 13, pp. 600–612, 2004.

[23] Z. Li, A. Aaron, I. Katsavounidis, A. Moorthy, and M. Manohara. "Toward a Practical Perceptual Video Quality Metric." Jun. 2016. [Online]. Available: http://techblog.net?ix.com/2016/06/toward-practical-perceptual-video.html

[24] F. Pescador, M. Chavarrías, M. J. Garrido, E. Juarez, and C. Sanz, "Complexity analysis of an HEVC decoder based on a digital signal processor," *IEEE Trans. Consum. Electron.*, vol. 59, no. 2, pp. 391–399, May 2013.

[25] D. Engelhardt, J. Moller, J. Hahlbeck, and B. Stabernack, "FPGA implementation of a full HD real-time HEVC main profile decoder," *IEEE Trans. Consum. Electron.*, vol. 60, no. 3, pp. 476–484, Aug. 2014.

[26] R. Garcia and H. Kalva, "Subjective evaluation of HEVC and AVC/H.264 in mobile environments," *IEEE Trans. Consum. Electron.*, vol. 60, no. 1, pp. 116–123, Feb. 2014.

[27] K. E. Psannis, "HEVC in wireless environments," *J. Real Time Image Process.*, vol. 12, pp. 509–516, Aug. 2016.

[28] J. Vanne, M. Viitanen, T. D. Hämäläinen, and A. Hallapuro, "Comparative rate-distortion-complexity analysis of HEVC and AVC video codecs," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 22, no. 12, pp. 1885–1898, Dec. 2012.

[29] E. Kalali and I. Hamzaoglu, "Approximate HEVC fractional interpolation filters and their hardware implementations," *IEEE Trans. Consum. Electron.*, vol. 64, no. 3, pp. 285–291, Aug. 2018.

[30] V. N. Dinh, H. A. Phuong, V. L. Cuong, and N. V. Thang, "Hardware-efficient and high-speed integer motion estimation architecture for HEVC," in *Proc. IEEE Int. Conf. Consum. Electron. Asia*, Oct. 2016, pp. 1–6.

[31] A. C. Mert, E. Kalali, and I. Hamzaoglu, "Low complexity HEVC sub-pixel motion estimation technique and its hardware implementation," in *Proc. IEEE 6th Int. Conf. Consum. Electron.*, Berlin, Germany, Oct. 2016, pp. 159–162.

[32] A. Mercat, F. Arrestier, H. Wassim, M. Pelcat, and D. Menard, "Energy reduction opportunities in an HEVC real-time encoder," in *Proc. IEEE Int. Conf. Acoust. Speech Signal Process.*, New Orleans, LA, USA, Mar. 2017, pp. 1158–1162.

[33] G. Corrêa, P. Assuncao, L. Agostini, and L. A. da Silva Cruz, "Complexity control of high efficiency video encoders," *IEEE Trans. Consum. Electron.*, vol. 57, no. 4, pp. 1866–1874, Nov. 2011.

[34] J. Sainio, A. Mercat, and J. Vanne, "uvgVenctester: Open-source test automation framework for comprehensive video encoder benchmarking," in *Proc. ACM Multimedia Syst. Conf.*, Istanbul, Turkey, Jun. 2021, pp. 255–260.

[35] A. Mercat, M. Viitanen, and J. Vanne, "UVG dataset: 50/120fps 4K sequences for video codec analysis and development," in *Proc. ACM Multimedia Syst. Conf.*, Istanbul, Turkey, May 2020, pp. 297–302.

[36] "FFmpeg." [Online]. Available: https://ffmpeg.org (Accessed: Jan. 20, 2022).

[37] J. Chen, Y. Ye, and S. Kim, "Algorithm description for versatile video coding and test model 13 (VTM 13)," Int. Telecommun. Union, Geneva, Switzerland, document JVET-S2002, ITU-T SG16 WP3, ISO/IEC JTC1/SC29/WG11, Apr. 2021.
[38] J. Chen, Y. Ye, and S. Kim, "Versatile video coding editorial refinements on draft 10," Int. Telecommun. Union, Geneva, Switzerland, document JVET-T2001, ITU-T SG16 WP3 and ISO/IEC JTC1/SC29/WG11, Oct. 2020.
[39] G. Bjøntegaard, "Improvements of the BD-PSNR model," Int. Telecommun. Union, Geneva, Switzerland, document VCEG-AI11, TU-T SG16/Q6, Jul. 2008.
[40] "Working practices using objective metrics for evaluation of video coding efficiency experiments," Int. Telecommun. Union, Geneva, Switzerland, document ITU-T HSTP-VID-WPOM, ISO/IEC DTR 23002-8, ITU-T SG16 WP3, ISO/IEC JTC1/SC29/WG11, 2020..
[41] B. Bross, J. Chen, J.-R. Ohm, G. J. Sullivan, and Y.-K. Wang, "Developments in international video coding standardization after AVC, with an overview of versatile video coding (VVC)," *Proc. IEEE*, vol. 109, no. 9, pp. 1463–1493, Sep. 2021.

Joose Sainio (Member, IEEE) received the M.Sc. degree in information technology from the Tampere University of Technology, Tampere, Finland, in 2018. He is currently pursuing the Ph.D. degree with the Ultra Video Group (UVG) of Tampere University of Technology.

He has been a part of UVG since 2016. His research interests include HEVC/VVC video coding, in particular enabling real-time encoding. He has experience in both hardware acceleration and more traditional optimization methods. Additionally, he has some familiarity with perceptual video coding and rate control.

Alexandre Mercat (Member, IEEE) received the M.Sc. and Ph.D. degrees in electrical and computer engineering from the Institut National des Sciences Appliquées of Rennes, Rennes, France, in 2015 and 2018, respectively.

He has been a Postdoctoral Researcher of Computing Sciences with Tampere University, Tampere, Finland, since 2018. His research interests include implementation of image and signal processing applications in many-core embedded systems, real-time implementations of the new generation video coding standards, complexity-aware video coding, machine learning, approximate computing, power consumption, and digital systems design. He received the Best Open Dataset and Software Paper Award from ACM MMSys'20 Conference.

Jarno Vanne (Member, IEEE) received the M.Sc. degree in information technology and the Ph.D. degree in computing and electrical engineering from the Tampere University of Technology, Tampere, Finland, in 2002 and 2011, respectively.

He is currently an Associate Professor with the Unit of Computing Sciences, Tampere University, Tampere. He is also the Founder and a Leader of the Ultra Video Group that is the leading academic video coding group in Finland. He has been the Project Manager for 22 international/national research projects. He is the author of over 80 peer-reviewed scientific publications. His research interests include HEVC/VVC video coding, ML-powered video coding, immersive 3D/360 media processing for extended reality, volumetric video capture and coding, vision-based environment perception in autonomous vehicles and drones, hybrid human–machine vision, remote machine control over 5G, telepresence, hardware accelerated video coding, video annotation, and virtual traffic simulation environments.