VVC Search Space Analysis Including an Open, Optimized Implementation

Adam Wieckowski, Jens Brandenburg, Benjamin Bross, *Member, IEEE*, and Detlev Marpe, *Fellow, IEEE*

***Abstract*—Versatile Video Coding (VVC) is a new video coding standard finalized in July 2020. During the standard development much attention was paid to keeping the decoding complexity increase as small as possible, with more permissive approach being taken with regard to the encoding. The VVC reference software VTM in random access configuration requires around double the time to decode and 8× the time to encode a video, compared to High Efficiency Video Coding (HEVC) reference software HM. With this runtime increase, an objective bitrate reduction of around 40% is achieved. In this paper we analyze the encoding complexity increase of VVC over HEVC. We abstract the implementation-based aspects, including low-level software optimizations and introduce an empirical measure to quantify the extent of encoder search space given a specific search algorithm. Based on the measure, we compare the search space of HM and VTM, but also of the open and optimized VVC encoder implementation VVenC, showing the potential for search space reduction and its impact on compression performance. Overall, it can be seen that while VVC's search space is quite large in VTM, it can be efficiently limited either by including early termination strategies or by disabling VVC coding tools.**

***Index Terms*—Optimization, video coding, video compression, VVC, search space.**

I. Introduction

THE VERSATILE Video Coding (VVC) standard [1] was finalized in July 2020. The project has been developed by the Joint Video Experts Team (JVET) of ITU-T VCEG and ISO/IEC MPEG. VVC is designed to achieve multiple objectives. First, the new standard should provide around 50% bitrate reduction at the same subjective quality over its predecessor, High Efficiency Video Coding (HEVC) [2], also jointly developed by the aforementioned bodies. Another objective is the applicability in versatile scenarios, ranging from conventional SDR video coding, through HDR, high-resolution, screen content coding, adaptive streaming to 360-degree video for virtual reality.

The coding efficiency improvement over HEVC has been officially tested in independent subjective evaluation. These verification tests [3], [4] confirmed an average bitrate reduction up to 50% for HD and UHD content and even more for 360-degree video.

The bitrate reduction comes at a cost of increased computational complexity. Measured in terms of runtime, in the commonly used random access configuration, VVC software test model VTM takes around 8× more time to encode and 2× more to decode than the HEVC software test model, HM [5]. First reports of optimized implementations including real-time decoder implementations [6]–[8] and optimized encoding algorithms [9]–[13] are available with very improved runtime. Those indicate that the VVC complexity, and its complexity increase when compared to HEVC, can only in limited manner be measured when comparing the reference software runtimes.

VVenC is an optimized encoder implementation [14], [15], initially released shortly after standard finalization and reaching version 1.0.0 in May 2021. The encoder can provide all of VTM's compression efficiency at less than half the runtime, and provides alternative presets constituting encoding time/efficiency tradeoffs.

Since the VVC standard only specifies the decoding process, the complexity of HEVC and VVC is defined by the number of operations required to decode a video from a conforming bitstream. A theoretical analysis backed by measured software runtime of available implementations [16] indicate the decoding process of VVC to be around two times more complex than HEVC. For the encoding, which is not standardized, but always depends on the particular implementation, the number cannot be quantified in a similar fashion. While the complexity of the decoding algorithms contributes directly to the encoding process, since an encoder always embeds a decoder, the process is usually very unsymmetrical with the encoder taking orders of magnitude more computational resources. While this discrepancy might be caused by the algorithms themselves, we assert it is foremost caused by the plurality of encoding options defining the encoding search space.

In this paper we want to capture and quantify the asymmetry between the algorithmic complexity of a standard mostly defined by the decoding process and the encoding search space, defined by the search algorithm of each specific encoder implementation. We propose a search space quantification method capturing the characteristics of block search differentiating between block subdivision search complexity, mode search within a block and combined search space. We discuss the presented complexity analysis approach comparing three encoder implementations – HM, VTM and VVenC at different presets representing different search algorithm variants.

Manuscript received June 30, 2021; revised January 18, 2022; accepted January 19, 2022. Date of publication February 3, 2022; date of current version June 3, 2022. *(Corresponding author: Adam Wieckowski.)*

The authors are with the Video Communication and Applications Department, Fraunhofer Institute for Telecommunications, Heinrich Hertz Institute, 10587 Berlin, Germany (e-mail: adam.wieckowski@hhi.fraunhofer.de; jens.brandenburg@hhi.fraunhofer.de; benjamin.bross@hhi.fraunhofer.de; detlev.marpe@hhi.fraunhofer.de).

Digital Object Identifier 10.1109/TCE.2022.3148813

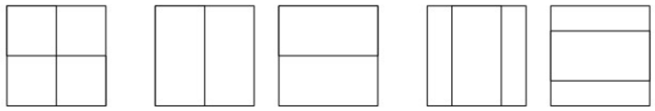


Fig. 1. Recursive splits available in VVC: quad-split, vertical and horizontal binary split, and vertical and horizontal ternary split.

The rest of this paper is structured as follows. In the next section an overview of hybrid block-based video coding with a special emphasis on the basics of HEVC and VVC is given. In Section III, the VVenC encoder is described. In Section IV, related work with regard to VVC complexity analysis is discussed. Section V introduces the empirical search space quantification method, while the measurements for three encoder implementations are discussed in Section VI. Section VII concludes the paper.

II. Block-Based Hybrid Video Coding

Most modern video codecs including HEVC and VVC are based on the block-based hybrid video coding paradigm. It has proven very efficient, while also allowing a straightforward definition of a deterministic decoding process, as required by the standardization.

In this paradigm, a video is first split into blocks, usually in a two-stage approach. First, a rigid subdivision into fixed size blocks is performed, e.g., macroblocks in H.264/AVC [17] and coding tree units (CTU) in HEVC and VVC. Then, each of the blocks can be further subdivided as specified by the block partitioning scheme of each standard. In AVC, a block subdivision can be selected per macroblock from a set of predefined models. In HEVC and VVC, a recursive subdivision based on different split modes is used. In HEVC, starting from a coding tree unit (usually containing 64×64 luma samples and the collocated chroma samples), a quad-split can be signaled indicating the block is to be split into four equally sized subblocks. This process is repeated recursively for each subblock until a predefined minimal block size is reached or it is signaled that no further split is performed. In VVC, the recursive split set is extended by four additional modes (Fig. 1), thus not only the split decision but also the split type has to be signaled. The leaves of the splitting process in VVC and HEVC are called coding units (CU).

For each coding unit, a coding mode has to be decided and signaled in the bitstream. In hybrid video coding, a prediction signal would be created for the block and the resulting residual error encoded and transmitted. The residual coding usually employs a quantization step, being the main source of distortion in lossy video coding. The prediction can be either based on a intra picture (using already coded samples from the currently coded picture) or inter picture (using coded samples from previously coded pictures) prediction. Within those two prediction modes, a specific mode has to be selected, e.g., intra directional mode, or block displacement for inter picture prediction (motion vector).

In HEVC, there is an additional split process available after the intra/inter decision, wherein each coding unit can be further non-recursively split into prediction units (PU). Each PU

TABLE I
Enumeration of Alternative Partitioning Configurations in HEVC Including Allowed PU Splits at Different Depths (Possibilities Only Available in Inter CUs Are Shown Cursive)

QT depth	CU size	Allowed PU splits
0	64×64	N×N, *N/2×N*, *N×N/2*, *N/2×N/2*, *N/4×N (L)*, *N/4×N (R)*, *N×N/4 (U)*, *N×N/4 (D)*
1	32×32	
2	16×16	
3	8× 8	N×N, *N/2×N*, *N×N/2*, N/2×N/2
Possible partitioning configurations per sample		
Intra	4	5
Inter	4	28

can utilize different prediction information, as long as all PUs within a CU all use either intra or inter picture prediction. The available PU splits for each available CU size are enumerated in Table I.

The main task for an encoder is to decide how to partition each CTU and which mode to use for each block. A naive encoder can go through all available options and decide the optimal encoding using the Lagrangian formulation of the rate-distortion optimization (RDO) problem:

$$\min\ \{J\},\ \text{where } J = D + \lambda \cdot R. \tag{1}$$

The coding cost J is a weighted average of distortion D and rate R, i.e., the number of bits required to represent a specific encoding. The value λ is derived from the desired encoding quality [18]. In (1), the resulting distortion has to be established, which usually requires applying the decoding process, thus the decoding complexity influence on the encoder.

The complexity of finding the optimal encoding through the optimization (1) is defined by the plurality of encoding modes for which the encoder estimates full coding cost J. In VVC, with its modular design and many CU level decisions being signaled in the bitstream [19], the plurality of those tested modes is much higher than in HEVC.

In modern encoders, the split search usually employs the so called G-BFOS (generalized Breiman, Freidman, Olshen and Stone) algorithm [20], [21]. The algorithm finds the optimal partitioning in a top down manner by iteratively deciding an optimal partitioning for a specific block. A split mode with k subblocks is treated as any other coding mode. Assuming the optimal coding costs for each ith subblock is defined as $J_{\mathrm{s,i}}$ and the rate of coding the split decision is given by R_{s} the coding cost of a split mode is defined as:

$$J_{\mathrm{s}} = \sum_{\mathrm{i}=1\cdots\mathrm{k}} J_{\mathrm{s,i}} + \lambda \cdot R_{\mathrm{s}}. \tag{2}$$

The optimal subblock cost $J_{\mathrm{s,i}}$ can itself be a result of splitting process (2). As can be seen in (2), with increasing split recursion depth, the number of decisions grows exponentially. In HEVC with only one split available the number of tested blocks increases with the depth, but the number of samples within a block decreases, effectively resulting in a linear relationship of maximal partitioning depth and overall number of processed samples, as will be shown further. For VVC, with multiple competing splits at each level [22], the growth is exponential again.

Within this search, the encoder can make decisions based on heuristics, i.e., without evaluating full coding cost (1) and (2). While this might lead to slight quality degradation, it usually significantly reduces the encoding complexity.

III. VVenC VVC Encoder

During the VVC standard development, software test model VTM has been used as a development platform and a proof of concept for the VVC standard. Albeit newly adopted algorithms had to show a reasonable tradeoff between coding gain and encoder runtime increase, software optimization has not been a distinct design goal of VTM.

A. VVenC Software Framework

The VVenC software encoder has been developed on top of a striped down version of VTM. This initial baseline contained only VVC partitioning, simple inter and intra prediction as well as the in-loop filters. All other tools were reimplemented, based on the VTM software. Consequently, many tools can produce equal results to their VTM counterparts. For some tools, e.g., multiple transform set (MTS) or intra sub-partitioning (ISP), alternative implementation approaches were used. In these cases, coding results will differ from VTM, yet achieving similar coding gains.

In VVenC a mixture of different techniques has been used to speed up the encoder. During the development and optimization of the software particular care has been taken to identify and remove redundancies in the algorithms and to optimize the memory layout to reduce performance bottlenecks. Based on an in-depth analysis the SIMD instruction vectorization has been revised and extended to support AVX2, and adding new SIMD code for quantization and transformation processing, affine prediction, loop filtering, motion compensated filtering, and more.

On top of the fine grained instruction level parallelization, a hybrid multi-threading scheme was adopted to the encoder software. This multi-threading scheme combines CTU and frame level parallelization, based on a single thread pool increasing efficient resource utilization and automatic workload balancing.

B. Encoder Search Space

Runtime improvements based on low-level optimizations, such as described in the previous section, are limited by a boundary, imposed by the inherent complexity of the algorithm. Part of this complexity can be attributed to an essential encoder component, the search for optimal encoding decisions. Instead of being a monolithic part, this search is the product of multiple hierarchical encoding decisions, establishing the overall encoder search space. Pruning this search space by reducing the number of encoding decisions provides the means to reduce the encoder complexity. Typically, such pruning leads to a degradation of coding performance. In [15], [23] we discuss several such pruning algorithms implemented in VVenC for different encoding decisions associated with tool

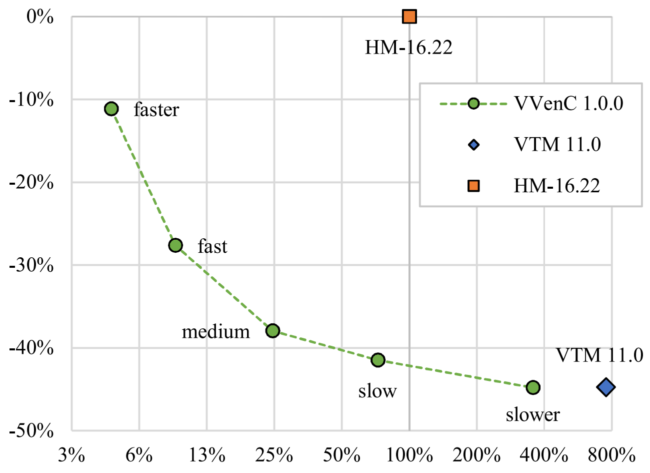


Fig. 2. Single threaded runtime and compression performance of HEVC and VVC reference encoders, HM and VTM, as well as an optimized VVC encoder VVenC (classes A1, A2 and B of JVET CTC in random access (RA) configuration [24]).

as well as non-tool specific algorithms, like motion estimation, partitioning search and others.

C. Pareto-Optimized Configuration Presets

Reducing the encoder complexity by pruning the search space for a certain tool or algorithm introduces a new tradeoff between encoder runtime and coding gain. To provide the full flexibility of VVC, these new tradeoffs have been exposed by additional encoder configuration options. Besides these VVenC specific configuration options, the encoder provides a large set of VTM-based options, also to adjust the coding efficiency. The full set of encoder configuration options constitutes a multi-dimensional vector. Each input configuration vector can be mapped onto a two dimensional operation point, representing a certain tradeoff between encoder runtime and coding efficiency. Finding optimal tradeoffs requires the determination of the set of Pareto optimal configuration vectors for which neither the coding gain nor the encoder runtime can be improved without impairing the other. For VVenC we used an iterative search algorithm to approximate a set of Pareto optimal configuration vectors. Based on those, five configuration presets are defined, representing different encoder operation points. According to the encoder runtime these presets are labeled faster, fast, medium, slow and slower, for ease of use. Overall, the coding gain of the presets spans from faster with 11% BD-rate gain over HM up to slower providing all of VTM's gain at less than half of its runtime. Comparing faster and slower presets against each other, a speedup of 70$\times$ can be achieved at the cost of reduced compression efficiency [23], showing improvement potential available by adapting the search space and algorithm.

The single-threaded compression performance vs runtime results, compared to HM and VTM are shown in Fig. 2. Each of the points constitutes an alternative search approach, utilizing different search space and traversing it in a different manner. Using the method proposed in Section V, we quantize and analyze the number of decisions contributing to the extent of the search space for each preset and contrast the numbers with VTM and HM.

IV. Related Work

Since the inception of the VVC standardization project, the complexity of the novel codec has been the topic of various publications. The literature varies with regard to the definition of complexity. In [16] the worst case decoding complexity is analyzed theoretically and by comparing the VVC reference software VTM to the HEVC reference HM as well as some early decoder implementations. The theoretical analysis assumes that a good approximation of the complexity is the number of performed multiply accumulate operations (MAC) and analyses core VVC algorithms accordingly. In the encoding complexity analysis [16], an early version of the optimized encoder VVenC is discussed. This paper extends this preliminary encoding complexity analysis with more details.

In [25]–[27] the encoding complexity is measured by the means of VTM reference software runtime, compared to that of HM. The analyses provide insightful information, each with a different focus. In [25], [26], [28] the runtime of different encoding aspects was measured and analyzed for different encoding quality settings. In [27], [29] the encoding time spent per video channel and block size in intra-only coding was emphasized. In [25], [29] the impact of code vectorization using SIMD is discussed. In [30] memory requirements of VTM and HM are analyzed during encoding process. As will be discussed further, the analyses based on the runtime and memory profiling of the reference softwares cannot abstract the different level of optimizations between the two encoders, but also between different encoder modules within each implementation.

Different approaches to complexity measurements were used in [15], [31], [32]. Instead of profiling the software with the full set of tools, the authors observed the software behavior when tools are turned off or their implementation (e.g., the search algorithm) is slightly adapted, up to enforcing optimal decisions without search in [31]. This way the overall complexity of the encoding process associated with a given tool can be quantified. Still, the analysis is only given in the context of a specific software implementation.

In this paper we try to separate the search space extent of a specific standard and the search sub-space a specific encoder actually evaluates from the computational complexity of specific signal processing algorithm, thus providing a level of implementation abstraction.

Many papers implicitly discuss algorithms concerning search space reduction, but use runtime as an evaluation criteria [9]–[13]. Such an approach is valid while comparing different search strategies within the same implementation, but cannot be applied to compare the search strategies of different encoders. Based on our proposed measure we discuss the search space increase between the reference softwares HM and VTM, and the search space reduction introduced in the optimized VVC encoder VVenC. Additional profiling results allow for more detailed VVC encoding complexity analysis.

V. Search Space Size Quantification

As concluded in [15], [16], [23], VVC complexity increase is in large part caused by the search space increase, i.e., the number of decisions the encoder has to make when selecting an optimal encoding for specific samples, possibly more so than the algorithmic complexity increase of specific algorithms. The latter can be somehow quantified by the decoding complexity increase, and the remaining disproportion between the encoding and decoding complexity increase can be roughly attributed to the increased number of encoder decisions.

We propose an approach to quantify the search space size for an encoder given a specific search algorithm. The presented approach is based on the assumption that the encoding search space of hybrid block-based video codecs is determined as the product of the partitioning search space and the number of tested coding modes per block. The former is determined by the flexibility of the partitioning framework implemented in a given codec. For VVC specifically, the novel quad-tree with binary and ternary tree framework allows multiple levels of recursive splits of different types, thus extending the search space in two dimensions – an encoder has to decide the split depth as well as split types. The mode search complexity is determined by the number of per-block signaled coding tools and modes. Specifically VVC, which was designed in a very modular fashion, contains many tools that can be selectively enabled or disabled on per-block basis. This concerns both the prediction, as well as residual coding modes available. We assume that the main burden of the coding unit search loop lies with the modes for which full RDO is done, i.e., for which a prediction is calculated, a residual formed and reconstructed residual as well as reconstruction error are calculated. We ignore the fact that many more prediction candidates are estimated without being passed on to full RDO.

In the following we define the encoding search space as the plurality of modes actually tested by the encoder. It is usually by orders of magnitude smaller than the product space of all possible encoding options allowed by a standard, and differs between encoders depending on the implemented search algorithm.

To quantify the two described main aspects contributing to search space size we define two empirical measures that combined estimate the search space size.

A. Partitioning Search Space

Assuming encoding of a frame of size $\mathrm{W} \times \mathrm{H}$ with 4:2:0 chroma sub-sampling, $N_{\mathrm{S}} = 1.5 \cdot \mathrm{W} \cdot \mathrm{H}$ samples are being encoded. During the CU search loop, the encoder decides for which partitioning configurations to perform a mode search. Assuming sequential operation, the encoder would perform a mode search in $i_{\mathrm{P}} \in 1 \cdots N_{\mathrm{P}}$ blocks. The luma and chroma size of each block are defined as $\mathrm{W}_{\mathrm{P}}(i, c)$ and $\mathrm{H}_{\mathrm{P}}(i, c)$, with $c = 0$ specifying luma size and $c = 1$ chroma size. Assuming neither U or V channel are ever tested without the other and

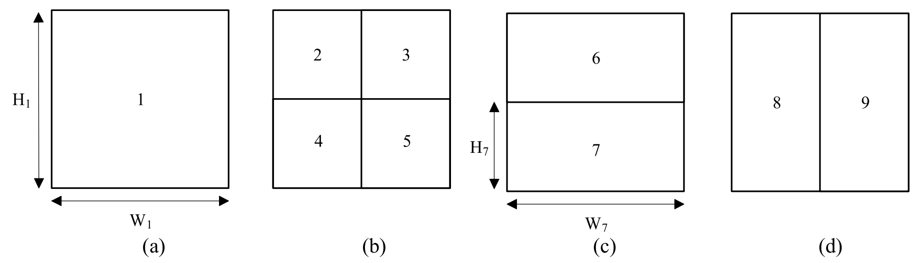


Fig. 3. Exemplary visualization of a block search of size 16 × 16 luma sample, with 4 alternatives evaluated (a)-(d). The block is analysed without subpartitioning (a), using a single quad-split (b), a horizontal split (c) and a vertical split (d), resulting in $N_S = 9$ subblocks for which mode search is performed. Ternary splits are not used in this example. Each subblock analysed in each of the partitioning configurations in sequentially enumerated as shown. $W_1 \cdots W_9$ and $H_1 \cdots H_9$ describe the blocks' widths and heights, with $H_1 = W_1 = W_6 = W_7 = H_8 = H_9 = 16$ and $W_{2 \cdots 5} = W_8 = W_9 = H_{2 \cdots 7} = 8$. $\mathrm{W_P}(i, c) = W_i \;/\; (1 + c)$ and $\mathrm{H_P}(i, c) = H_i \;/\; (1 + c)$, as used in (3).

allowing $\mathrm{W_P}(i, c)$ and $\mathrm{H_P}(i, c)$ to be equal to zero, e.g., in the case of Chroma Separate Tree coding in VVC, the partitioning search space during encoding can be defined as (see Fig. 3):

$$S_P = \sum_{c=0 \cdots 1, i=1 \cdots \mathrm{Np}} \mathrm{W_P}(i, c) \cdot \mathrm{H_P}(i, c) \cdot (1 + c)/N_S. \quad (3)$$

The partitioning search space (3) quantifies how often a sample is tested within different partitioning configurations. Assuming the encoder cannot encode a sample without performing a CU search within a valid partitioning it can be assumed that $S_P \geq 1$. In the special case of $S_P = 1$, the encoder does not test alternative partitioning configurations with each sample being tested in exactly only one partitioning configuration (e.g., at the CTU level only without any further splitting or given a-priori knowledge of optimal partitioning [31]). Given the G-BFOS algorithm on which the search in HM, VTM and VVenC are based and assuming no block is being visited multiple times in the same partitioning configuration during the search, an upper bound on S_P can be formulated by enumerating all possible partitioning layouts within a search. The numbers will be derived and discussed in the next section for VVC and HEVC.

B. Coding Mode Search Space

We define the coding modes search space as the number of modes for which full RDO (1) including quantization step is performed. The rationale behind that comes from relatively high complexity of the quantization task, more so if rate-distortion optimized quantization is being performed.

Analogue to Section V-A, the mode search space is the average number of quantization processes each to-be-encoded sample is part of, normalized by S_P and the number of samples in frame N_S. Assuming the encoder performs the quantization $i_Q = 1 \cdots N_Q$ times, the size of the to be quantized residual in the i_Qth block is defined as $\mathrm{W_Q}(i) \cdot \mathrm{H_Q}(i)$. Since it is assumed that the quantization is done for only a single component at a time, it is not required to differentiate between the components. The mode search space (or quantization search space), is defined as:

$$S_Q = \sum_{i=1 \cdots \mathrm{Nq}} \mathrm{W_Q}(i) \cdot \mathrm{H_Q}(i) \;/\; (N_S \cdot S_P). \quad (4)$$

Contrary to S_P, S_Q (4) can be smaller than one, e.g., in a case the encoder is forcing no-residual coding within a frame, e.g., due to pre-analysis-based decision or time constraints. It is not possible to formulate an upper bound for S_Q, since the number of full RDO processes lies fully in the encoder search algorithm domain and is not strictly bound by any normative codec constraint, as in the case of partitioning. In the worst case, an encoder could perform full RDO in each step of motion search, instead on performing the motion search based on distortion only and do full RDO for the final motion estimation candidate only.

C. Combined Search Space

The combined search space, i.e., the multiplier of additional decisions the encoder has to take compared to encoding with all decision know a-priori, can be defined as:

$$S = S_P \cdot S_Q. \quad (5)$$

In the following sections the complexity analysis of VVC will be performed based on measures S_P, S_Q and S. Measure S estimates, based on empirical data, how often a full per-block encoding and decoding process has to be performed for each sample.

The described measures S_P, S_Q and S are independent of the actual software or hardware implementation, including low-level optimization, but they depend on the used search algorithm.

The measurement of the overall coding block search space through the measurement of the number of the evaluated partitioning and quantization blocks per sample is motivated by the two aspects wrapping the CU search. The partitioning defines for which blocks mode search is to be performed. Quantization is a computationally intensive step of full RD-cost being collected for a given block, indicating number of modes which are tested. The normalization by the number of encoded samples is motivated by reduced complexity of small block encoding, even if the relationship is not linear (small blocks are more costly to process per sample [16]).

VI. VVC Complexity Analysis

A. Experimental Setup

The following experimental data was acquired using versions of VVenC 1.0.0, VTM-11.0 and HM-16.22 modified

TABLE II
EMPIRICAL QUANTITATIVE SEARCH SPACE MEASUREMENTS FOR TWO VVC ENCODERS AND THE HEVC REFERENCE MODEL HM FOR I FRAMES ENCODING

	Partitioning S_P				**Quantization S_Q**			**Combined S**		
	avg	**min**	**max**	**bound**	**avg**	**min**	**max**	**avg**	**min**	**max**
VVenC faster	3.43	2.84	4.12	4.67	4.57	4.48	4.63	15.69	12.71	19.10
VVenC fast	6.51	5.03	8.29	14.33	13.64	13.42	13.82	88.71	67.53	114.62
VVenC medium	12.12	8.49	16.80	37.50	19.38	18.76	19.90	234.95	159.22	334.36
VVenC slow	23.55	14.67	36.43	85.42	18.26	18.04	18.49	429.89	264.65	673.59
VVenC slower	28.34	17.20	44.89	85.42	28.22	27.57	29.07	799.82	474.34	1304.79
VTM-11.0*	29.68	19.33	45.89	85.42	22.59	22.30	23.05	670.39	437.76	1071.62
HM-16.22 CU	3.95	3.95	3.95	4.00	10.36	10.22	10.53	40.89	40.35	41.57
HM-16.22 CU+PU	4.95	4.95	4.95	5.00	7.73	7.64	7.84	40.89	40.35	41.57

TABLE III
EMPIRICAL QUANTITATIVE SEARCH SPACE MEASUREMENTS FOR TWO VVC ENCODERS AND THE HEVC REFERENCE MODEL HM FOR P/B FRAMES ENCODING

	Partitioning S_P				**Quantization S_Q**			**Combined S**		
	avg	**min**	**max**	**bound**	**avg**	**min**	**max**	**avg**	**min**	**max**
VVenC faster	2.39	1.71	3.31	5.00	3.15	2.92	3.55	7.32	5.00	11.74
VVenC fast	2.34	1.68	3.23	5.00	4.46	4.13	5.05	10.46	6.92	16.31
VVenC medium	4.69	3.32	6.75	24.00	5.81	5.29	6.59	27.26	17.59	44.49
VVenC slow	8.98	6.01	14.12	75.00	10.08	9.37	10.99	90.52	56.30	155.13
VVenC slower	23.71	13.49	42.04	220.75	13.08	11.85	14.84	310.17	159.92	623.73
VTM-11.0*	29.18	17.54	49.60	220.75	14.28	12.35	16.91	416.51	211.83	852.12
HM-16.22 CU	3.95	3.95	3.95	4.00	20.93	17.18	27.27	82.60	67.83	107.62
HM-16.22 CU+PU	15.51	13.97	17.77	28.00	5.32	4.84	6.05	82.60	67.83	107.62

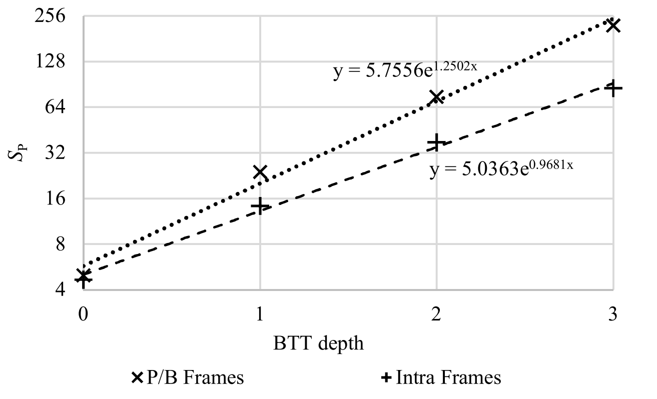


Fig. 4. Approximate exponential relationship between BTT-depth and partitioning search space S_P bound in VVC, including a least-squares approximation.

to measure S_P and S, from which $S_Q = S/S_P$ is derived. The data was acquired for HD and UHD sequences from the JVET common test conditions (classes A1, A2, and B) [24], analogue to the results presented in Fig. 2. For VTM-11*, the presented data results from encoding a reduced set of only 100 frames due to long encoding times. Additionally, a low-overhead profiling was enabled in VVenC 1.0.0 to measure runtime of different encoding aspects (using the ENABLE_PROFILING macro). This low-overhead profiling tool was ported to VTM-11.0 and HM-16.22 to acquire comparable timings. The profiling time measurements in Fig. 4 were performed on 32 core server blades with Intel Xeon E5-2697A v4 @2.6GHz CPUs, in single-threaded encoder operation.

B. Search Space Analysis

Tables II and III summarize the empirical search space measures S_P, S_Q and S for the five presets of VVenC, VTM in CTC configuration, as well as HM in VVC CTC configuration. For HM, two alternative approaches to dealing with prediction unit splits are given, with the “HM-16.22 CU” showing the number when not treating PU splits as a part of the partitioning framework. Since the VVC standard does not allow PU splitting in a sense it is used in HEVC, additionally an alternative measurement “HM-16.22 CU+PU” is shown, in which the PU splits are accounted for in the S_P partitioning search space measurement, rather than S_Q. The mode constraint concept of VVC is somehow similar to the HEVC PU concept, even if motived differently, and for VVC it is also accounted for in S_P. The combined search space S should be equal for both HM measurements, with interpretation of PU splits shifting the search space between partitioning and modes search. For further discussions, “HM-16.22 CU+PU” will be chosen as baseline if not explicitly stated differently.

The min and max measurements constitute the minimal and maximal geometric mean over the sequences at specific QP, showing the value span between encoding at different quality settings.

1) Partitioning Search Space: Additionally to the measured values, an upper bound for S_P, assuming an encoder using the G-BFOS top-down search of all possible blocks without early terminations, is derived. For VVC, it is derived by counting all possible blocks within a CTU search without any early termination strategies. This way, it can be ensured

TABLE IV
VVC SEARCH SPACE BOUND DEFINING PARTITIONING HIGH-LEVEL PARAMETERS USED IN VVENC PRESETS AND VTM

	Intra Frames						P/B Frames					
	CTU size	Max size		Max depth		S_P bound	CTU size	Max size		Max depth		S_P bound
		QT	BTT	QT	BTT			QT	BTT	QT	BTT	
VVenC faster	64	64	N/A	4	0	4.67	64	64	N/A	4	0	5.00
VVenC fast	64	64	32	4	1	14.33	64	64	N/A	4	0	5.00
VVenC medium	128	128	32	4	2	37.50	128	128	128	4	1	24.00
VVenC slow	128	128	32	4	3	85.42	128	128	128	4	2	75.00
VVenC slower	128	128	32	4	3	85.42	128	128	128	4	3	220.75
VTM-11.0	128	128	32	4	3	85.42	128	128	128	4	3	220.75

that all concepts and constraints can be accounted for in the partitioning, including VPDU, global and local separate trees and mode constraints. For HEVC, and thus the search in HM, the derivation is more straightforward – each sample can be visited within a block at a depth between 0 and 3 corresponding to block sizes of 64 × 64 down to 8 × 8, accounting to maximum four possible partitionings being tested for each sample. In case of the measurement including the PU splits, at each of the depths, any of the possible PU splits at each depth constitutes a partitioning alternative, as enumerated in Table I.

The HM search algorithm does not include any early termination conditions in the CU partitioning. For this reason, the minimum and maximum values are the same in that case and almost equal the upper bound. The small difference is caused by picture boundary encoding, at which some blocks available in partitioning extend beyond encoded samples and will be further split without a CU search in the specific block. In HM, there is no early termination condition for the N/2 × N/2 intra split, meaning that the previous statement holds for the intra frames even if accounting for the PU split. For P and B frames, the PU split search for inter predicted blocks contains some speedup logic, resulting in around 50–65% of the available search space being visited (min. and max. values from Table III, respectively).

As previously discussed and widely agreed in literature [9], [10], [12], [13], the partitioning in VVC is much more complex with more partitioning depths available and multiple split options at each level. Comparing the upper bound of the partitioning search space S_P, and assuming the G-BFOS algorithm, this complexity increase can be quantified in terms of search space size. It has to be noted that for both HEVC and VVC, the extent of this search space is defined by high-level parameters, like CTU size (partitioning root) and maximal allowed depth or block size constraint. The number for HM, due to its exhaustive configuration, constitutes the worst case for HEVC. On the other hand, the numbers for the VVC encoders VTM and VVenC constitute various predefined working points defined by CTC configuration for VTM [24] and Pareto optimization for VVenC [15], [23]. The search space could be considerably extended, e.g., by allowing more than three recursive BTT splits for VTM. The summary of used high-level partitioning parameters defining VVC partitioning search space bound is shown in Table IV.

For VTM the search space is increased around 17× for intra frames and around 8× for P/B frames over HM (or respectively over 20× and 55×, if not accounting for the PU split in HEVC).

As described in [9], VTM already employs a set of very effective fast partitioning search strategies. The described geometric average speedup of 2.5× for all intra and 7× for random access [9] is in line with the obtained numbers of visited blocks per sample compared to the upper bound (25-60% for intra frames and 7-22% for P/B frames), indicating that the extent of actually visited search space directly translates into encoding time. Using the G-BFOS algorithm, the exponential increase in possible partitioning configurations with increasing depth in HEVC translates into a linear increase of search space S_P, due to the per-sample normalization. In VVC, the partitioning is not only defined by the maximal allowed depth of the single QT split, but also by the availability of alternative splits, making the partitioning search space S_P bound increase exponentially with regard to the maximal allowed number of recursive alternative splits, as can be observed in Fig. 4, even when using the G-BFOS approach.

Due to the exponential search space increase, controlling the maximal BTT depth partitioning parameter with regard to the used working point is of crucial importance. In VVenC the partitioning parameter are derived in an iterative Pareto optimization approach together with all other coding options [15]. For this reason, only the slower configuration utilizes the VTM partitioning configuration, while in all other presets the intra or inter frame maximal depth, or both, are reduced. This results in the search space successively decreasing towards faster, where the upper bound roughly resembles the "HM-16.22 CU" numbers. VVenC faster, similar to HEVC, only allows quad-splits. In VVenC fast and faster the additional QT depth of 4 with 4 × 4 CU size is allowed, which is not available in HEVC. The upper bound of S_P for VVenC medium is in the range of the "HM-16.22 CU+PU" value. Utilizing the optimized partitioning search, S_P value is 3–4× smaller than the bound for this preset.

On top of the search space restrictions introduced by limiting the number of allowed recursive splits, VVenC employs a set of additional heuristics in partitioning search [10]. Combined with reduced configuration, in the presets faster through slow VVenC actually visits less partitioning candidates than HM, which is consistent with it being faster than HM in those configurations (compare Fig. 2). In the slower preset S_P is still noticeably reduced by between 5-25% in comparison to VTM through an adapted heuristic [10].

The relatively large span of S_P in each encoder shows that many of the partitioning early exit heuristics are content dependent and will mostly improve low-bitrate or low-variance content encoding. Towards VVenC in faster presets, this discrepancy reduces indicating that with reduced number of possible encodings (S_P bound), its harder to find effective early termination heuristics [10].

2) Coding Mode Search Space: After an encoder decides to start a CU search within a specific block, increasing the measure S_P, a set of coding modes for that specific block is tested to calculate or estimate their RD-cost to decide a specific optimal encoding for that given block. Within this calculation, the quantization process has been chosen to represent a quantitative measure for the extent of the per-block search space including alternative prediction and residual coding modes that can be signaled per block and need to be tested to find the coding optimum.

Comparing the HM and VTM numbers shows that in VTM CTC configuration, VVC introduces additional complexity. VVC has more mode alternatives available per block, with more intra modes, merge candidates, other improvements to motion information signaling and residual generation and coding alternatives including new transformation types. HEVC on the other hand includes the concept of residual quad-tree (RQT) which increases the number of quantization processes per block. The TU split used in RQT is counted as a part of S_Q rather than S_P. Overall, S_Q is around 2–3$\times$ larger for VTM than HM, similar to the S_P increase for P/B frames, showing a balanced increase between coding tools and partitioning in motion compensated encoding. For intra frame coding, the 2–3$\times$ increase of S_Q is much smaller than the around 10$\times$ increase of S_P, showing an imbalance.

Contrary to the partitioning search, the mode search including RQT decisions in HM is fairly optimized, containing many pruning strategies. The impact of the interpretation of the PU splits either as a part of partitioning or mode search indicates it has a big overall impact on the HM encoder search, roughly similar to the partitioning without PU splits itself.

As previously discussed, to achieve the compression efficiency of VTM, VVenC slower utilizes a similar search algorithm, with the 2$\times$ speedup being more attributed to improved implementation. For this reason, just as S_P, S_Q is similar for both VTM and VVenC slower. It is reduced gradually with the introduction of additional fast algorithms including early search terminations and disabling some encoding tools along the Pareto Set towards faster configurations. For intra frames, the mode search in VVenC slower is actually more extensive than in VTM, increasing S_Q by around 20%. This can be attributed to alternative implementation of some residual coding tools and the random access based Pareto optimization, which tends to obscure the runtime impact of intra coded frames.

In VVenC medium, the extent of S_Q in P/B frames is similar to HM, indicating that the 4$\times$ speedup vs HM in this configuration is mostly attributed to improved partitioning search and implementation.

As discussed in Section III, VVenC faster mostly includes coding tools not requiring per-block signaling, i.e., not increasing S_Q. This way, on average only around three different sets of residuals are being quantized for full RD-cost calculation in P/B frames and 4.5 in intra frames. The 2$\times$ runtime increase from VVenC faster to fast is larger than the increase of per-block search space. It is attributed more to the non-CU tools, like additional loop filter ALF and the application of a temporal filtering, none of which is captured by the measurement of S_P, S_Q and S.

C. Overall Complexity

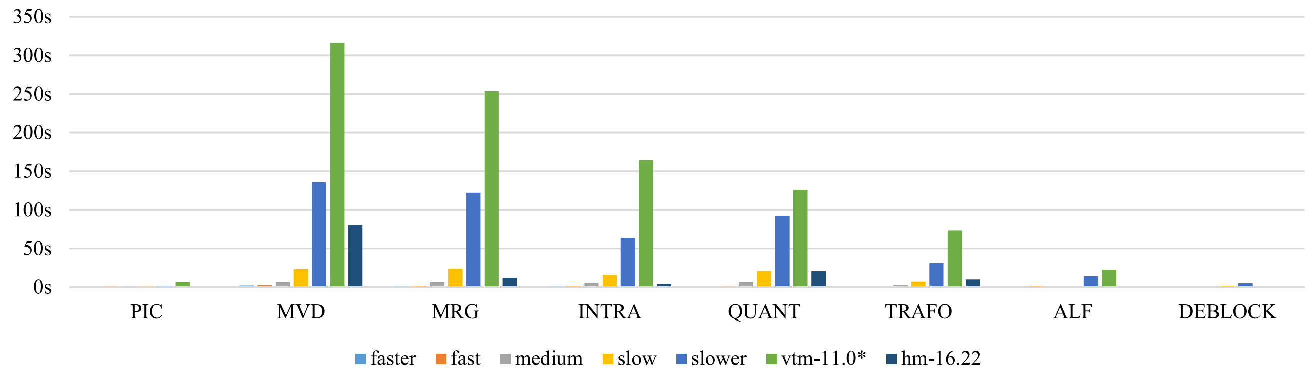
Complementary to the search space increase, VVC introduces new complex normative algorithms, determining the decoding complexity and thus having a major impact on the encoding complexity. Examples of such algorithms include more complex merge derivation algorithms, template-based residual coding CABAC context derivation, decoder side motion vector refinement and a new pipeline step, the adaptive loop filter.

The decoding complexity increase has been throughout studied in the standardization process [32] and was analyzed both empirically and theoretically [16]. It is broadly agreed that VVC, utilizing all coding tools, requires around double the computing power to decode over HEVC. This number also roughly resembles the decoding time increase between HEVC reference decoder HM and VVC reference decoder VTM, both of which are optimized to a similar degree. Faster software decoders are available for both standards showing optimization potential. VVdeC, a freely available software decoder for VVC, allows more than 2$\times$ speedup over VTM without multi-threading, utilizing optimized tool implementations and more throughout vectorization. Similar speedup is achieved between VTM encoder and VVenC in slower preset, roughly showing the impact of more optimized implementation in the latter.

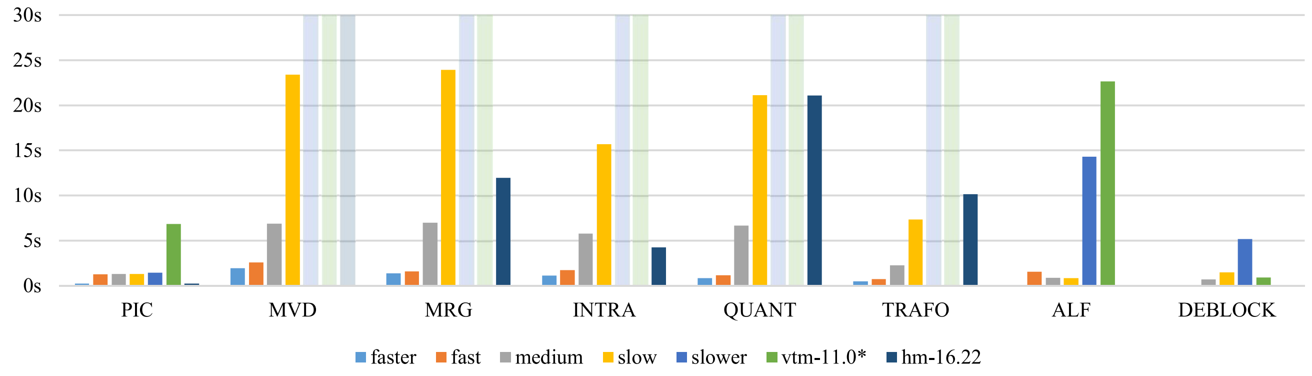
The reported encoder runtime increase of VTM over HM is around 8$\times$. With the implementation complexity accounting for around 1.5-2$\times$, the remaining increase of 4-5.5$\times$ has to be attributed to search space extension, which is confirmed in the search space analysis in Table III (for random access configuration complexity is mostly determined by P/B frames).

Fig. 5. shows average per-frame profiling results of VVenC, VTM and HM, accumulated as a geometric mean across all tested sequences. While the profiling shows the combined effect of implementation complexity as well as search space increase, it still allows to draw some conclusions. In the following analysis the acronyms are used as defined in [31].

1) Motion Search: Time spent in motion search (MVD) in VTM is around 4$\times$ larger than in HM. For VTM though, motion vector difference search consists of multiple parts. Alternative motion models including affine motion and adaptive motion vector resolution each constitute separate encoder search paths. Both models are new in VVC and can be signaled



(a) Full scale.



(b) Y-Axis span scaled for VVenC presets and HM.

Fig. 5. Low-overhead profiling results for VVenC 1.0.0 in 5 presets, VTM-11 and HM-16.22. The instrumentation results are normalized to the number of encoded frames and averaged across the tested sequences and QP points using a geometric mean.

TABLE V
PEARSON AND SPEARMAN RANK CORRELATION OF RUNTIME AND COMBINED SEARCH SPACE S. THE CORRELATION IS FAIRLY HIGH WITHIN A SPECIFIC IMPLEMENTATION, WITH THE HIGHEST SCORES REACHED IN LESS OPTIMIZED ENCODERS (VTM AND HM) AS WELL AS CONFIGURATIONS WITH MOST OF THE COMPLEXITY CONTAINED WITHIN THE CU-LOOP (FASTER AND HM). LAST COLUMN SHOWS COMBINED RESULTS FOR TWO IMPLEMENTATIONS

Resolution	**Correlation**	**faster**	**fast**	**medium**	**slow**	**slower**	**vtm-11***	**hm-16.22**	**slower and vtm-11**
HD	**Pearson**	0.95	0.91	0.85	0.84	0.89	0.93	0.97	0.91
	Spearman	0.97	0.94	0.89	0.89	0.91	0.95	0.96	0.90
UHD	**Pearson**	0.93	0.89	0.90	0.87	0.91	0.93	0.95	0.90
	Spearman	0.97	0.92	0.94	0.91	0.95	0.96	0.92	0.92

per block thus attributing to search space expansion rather than complexity increase.

2) Merge Candidate Search: The profiling of merge candidate search indicates similar conclusions. The MRG bar for VTM is an order of magnitude larger than for HM, which is caused by the increased number of merge candidate derivation and application methods. With SMVD and MMVD the initial merge candidate is further extended, GPM allows blending of multiple merge candidates, all increasing the number of tested prediction candidates and possibly residuals. Only the DMVR and BDOF tools, being implicitly applied, add to the complexity of final motion compensation once a merge candidate is decided without extending the search space. Both of those implicit tools are active in VVenC faster, and it can be seen that the profiling results still indicate low runtime compared to HM.

3) Picture Level Processing: The increased picture level processing time in VTM and VVenC are attributed to the motion compensated temporal filtering step, which in VVenC is further optimized using vectorization.

4) Deblocking: The deblocking filter in HM can be applied very efficiently, contributing no measurable time to the overall encoding. This is different for VTM and caused by the increased deblocking filter resolution and the introduction of subblock motion models requiring fine deblocking even for large block sizes. While the deblocking implementation is improved in VVenC, this is the only encoding aspect VVenC spends more time on than VTM. This behavior is caused by the inclusion of the deblocking refinement into the CU search loop. This additional step increases processing time of each test without increasing the number of tested modes in VVenC.

5) Adaptive Loop Filter: When comparing the results of different VVenC presets, implementation differences don't play a role in the profiling. Starting from VVenC slower, comparing the bars towards VVenC faster, the profiling results indicate search space reduction. Especially interesting in this regard is the runtime reduction of ALF from VVenC slower to slow, attributed to the exclusion of clipping values search, reducing the search space by $16\times$, which directly translates to respective runtime reduction.

This processing step, just like previous two, happens outside of the CU search loop and is not captured by the search space measurements, showing the limitations of the proposed method.

6) Other Observations: The difference between profiling results of VTM and VVenC slower, apart from deblocking, indicate the added level of optimization for each given processing stage. Apart from the temporal filter implementation, especially motion search, residual transformation and intra prediction were efficiently improved, the latter two of which have no vectorization included in VTM.

7) Correlation Analysis: Table V shows the Pearson and Spearman Rank correlation of the combined search space S (weighted sum of I and P/B frame values, using the intra period as weight factor) and overall per-frame runtime. Each sequence at each tested QP represents a data-point. Especially the Spearman Rank correlation represents if the order of data is similar, ignoring the linearity of the relationship. It is apparent that the correlation is fairly high overall, but the highest for less optimized encoders and configurations with most of the workload contained within the measured CU-loop (VVenC faster). Lower correlation in more optimized VVenC indicates increased discrepancy between the search space and runtime, as the different complexity of various decoding processess required to evaluate an encoding cost is amplified.

Analyzing the combined correlation of VVenC slower and VTM in Table V, which utilize a similar encoding algorithm but within different software implementation, the combined Spearman Rank correlation is lower than in each encoder by itself. This can be attributed to differences in the implementation. Alas, the trellis quantization algorithm implementation, making up large portion of the runtime, is comparable in VVenC and VTM, explaining the relatively low magnitude of the difference.

8) Summary: Overall, it can be seen that while the improved implementation allows VVenC slower to be more than $2\times$ faster than VTM, the search space reduction allows for more impactful runtime reduction, alas at the cost of compression efficiency as shown in Fig. 2.

VII. Conclusion

In this paper we discussed the VVC encoding runtime increase over HEVC, as can be observed comparing the reference implementations VTM and HM. We defined an empirical measure for encoder search space size quantification and measured it for HM, VTM and an optimized VVC encoder VVenC. The measure can be used to compare encoding search algorithm complexity not in the terms of runtime, but rather in terms of performed rate-distortion tests per coded sample, thus abstracting implementation and underlying hardware. We propose to measure the coding unit loop search space extent in terms of partitioning and mode overhead. For partitioning the number of times a sample is part of a block search is counted. The plurality of per-block mode decisions is defined as the number of quantization processes happening within a block. The product of the two measurements defines the combined complexity of the coding unit search loop.

For the partitioning search space, upper bound is defined for specific high-level configurations, estimated theoretically in HEVC and empirically in VVC. It can be observed that the partitioning as configured in VTM is one of the biggest contributors to VTM search space. In most VVenC presets, the partitioning search space bound is largely reduced by decreasing the number of allowed recursive splits, while additional heuristic algorithm ensure the actually measured partitioning search space S_P is smaller than the bound.

We showed that with its modular design VVC offers the encoder many possibilities, effectively increasing the number of encoder decisions, when compared to HEVC. Some of the new tools do not require encoder search, and their complexity is reflected in decoding complexity increase, as analyzed in other literature.

We compared different encoding algorithm variants in VVenC and VTM. Without significantly reducing the search space and impairing the compression efficiency, VVenC provides similar gains over HM at less than half the runtime. Still, the biggest speedup potential lies in search space reduction either by the means of reducing the available toolset or introducing early exits. In VVenC, those options are mixed during the configuration space Pareto optimization. While this comes at compression efficiency tradeoff, the resulting working points still provide significant gain over HM, which employing a fairly thorough search algorithm can be seen as approximate upper bound for HEVC efficiency. Most VVenC presets employ a smaller search space than HM, which is reflected in reduced runtime.

In the future, a quantification of the prediction search space can be measured in a similar manner, e.g., by counting the samples processed in distortion computation by an encoder, normalized by the overall number of encoded samples. Comparing such a measure to the mode search space S_Q could provide insights into the degree of optimization of an encoder and the amount of implemented early exits.

Overall, VVC provides a large toolset that the encoder can select from depending on user needs and application constraints. This paper demonstrates that while VVC offers an encoder more decisions than HEVC, it does not force it to evaluate them. Selecting the most promising paths from a wide set, as VVenC does, provides better BD-rate gain much faster than looking into every last corner as in the case of HM or VTM (or, VVenC slower).

References

[1] "Versatile video coding," ITU, Geneva, Switzerland, ITU-T Recommendation H.266, Aug. 2020.

[2] "High efficiency video coding," ITU, Geneva, Switzerland, ITU-T Recommendation H.265, Apr. 2013.

[3] V. Baroncini and M. Wien, *VVC Verification Test Report for UHD SDR Video Content*, document JVET-T2020 of ITU-T/ISO/IEC Joint Video Experts Team (JVET), JVET, Oct. 2020.

[4] V. Baroncini and M. Wien, *VVC Verification Test Report for HD SDR and 360° Video Content*, document JVET-V2020 T2020 of ITU-T/ISO/IEC Joint Video Experts Team (JVET), JVET, Apr. 2021.

[5] F. Bossen, X. Li, K. Sühring, K. Sharman, and V. Seregin, *JVET AHG Report: Test Model Software Development (AHG3)*, document JVET-V0003 of ITU-T/ISO/IEC Joint Video Experts Team (JVET), JVET, Apr. 2021.

[6] A. Wieckowski *et al.*, "Towards a live software decoder implementation for the upcoming versatile video coding (VVC) codec," in *Proc. IEEE Int. Conf. Image Process. (ICIP)*, Abu Dhabi, UAE, Oct. 2020, pp. 3124–3128, doi: 10.1109/ICIP40778.2020.9191199.

[7] B. Zhu *et al.*, "A real-time H.266/VVC software decoder," in *Proc. IEEE Int. Conf. Multimedia Expo (ICME)*, Shenzhen, China, 2021, pp. 1–6, doi: 10.1109/ICME51207.2021.9428470.

[8] F. Bossen, *Performance of a Reasonably Fast VVC Software Decoder*, document JVET-S0224 of ITU-T/ISO/IEC Joint Video Experts Team (JVET), JVET, Jun./Jul. 2020.

[9] A. Wieckowski, J. Ma, H. Schwarz, D. Marpe, and T. Wiegand, "Fast partitioning decision strategies for the upcoming versatile video coding (VVC) standard," in *Proc. IEEE Int. Conf. Image Process. (ICIP)*, Taipei, Taiwan, 2019, pp. 4130–4134, doi: 10.1109/ICIP.2019.8803533.

[10] A. Wieckowski, B. Bross, and D. Marpe, "Fast partitioning strategies for VVC and their implementation in an open optimized encoder," in *Proc. Picture Coding Symp. (PCS)*, 2021, pp. 1–5.

[11] M. Aklouf, M. Leny, F. Dufaux, and M. Kieffer, "Low complexity versatile video coding (VVC) for low bitrate applications," in *Proc. 8th Eur. Workshop Vis. Inf. Process. (EUVIP)*, Rome, Italy, Oct. 2019, pp. 22–27.

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

[13] N. Tang *et al.*, "Fast CTU partition decision algorithm for VVC intra and inter coding," in *Proc. IEEE Asia–Pacific Conf. Circuits Syst. (APCCAS)*, Bangkok, Thailand, 2019, pp. 361–364.

[14] A. Wieckowski *et al.*, "VVenC: An open and optimized VVC encoder implementation," in *Proc. IEEE Int. Conf. Multimedia Expo Workshops (ICMEW)*, Shenzhen, China, 2021, pp. 1–2, doi: 10.1109/ICMEW53276.2021.9455944.

[15] J. Brandenburg *et al.*, "Towards fast and efficient VVC encoding," in *Proc. IEEE 22nd Int. Workshop Multimedia Signal Process. (MMSP)*, Tampere, Finland, 2020, pp. 1–6, doi: 10.1109/MMSP48831.2020.9287093.

[16] 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, doi: 10.1109/TCSVT.2021.3072204.

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

[18] G. J. Sullivan and T. Wiegand, "Rate-distortion optimization for video compression," *IEEE Signal Process. Mag.*, vol. 15, no. 6, pp. 74–90, Nov. 1998.

[19] B. Bross *et al.*, "Overview of the versatile video coding (VVC) standard and its applications," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 31, no. 10, pp. 3736–3764, Oct. 2021.

[20] D. Marpe *et al.*, "Video compression using nested quadtree structures, leaf merging, and improved techniques for motion representation and entropy coding," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 20, no. 12, pp. 1676–1687, Dec. 2010.

[21] G. J. Sullivan and R. L. Baker, "Efficient quadtree coding of images and video," in *Proc. Int. Conf. Acoust. Speech Signal Process. (ICASSP)*, vol. 4. Toronto, ON, Canada, 1991, pp. 2661–2664, doi: 10.1109/ICASSP.1991.150949.

[22] 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, doi: 10.1109/TCSVT.2021.3088134.

[23] J. Brandenburg, A. Wieckowski, A. Henkel, B. Bross, and D. Marpe, "Pareto-optimized coding configurations for VVenC, a fast and efficient VVC encoder," in *Proc. IEEE 23rd Int. Workshop Multimedia Signal Process. (MMSP)*, 2021.

[24] F. Bossen, J. Boyce, X. Li, V. Seregin, and K. Sühring, *VTM Common Test Conditions and Software Reference Configurations for SDR Video*, document JVET-T2010 of ITU-T/ISO/IEC Joint Video Experts Team (JVET), JVET, Oct. 2020.

[25] Í. Siqueira, G. Correa, and M. Grellert, "Rate-distortion and complexity comparison of HEVC and VVC video encoders," in *Proc. IEEE 11th Latin Amer. Symp. Circuits Syst. (LASCAS)*, 2020, pp. 1–4, doi: 10.1109/LASCAS45839.2020.9069036.

[26] 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. (ICIP)*, Abu Dhabi, UAE, 2020, pp. 3134–3138, doi: 10.1109/ICIP40778.2020.9190983.

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

[28] 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, doi: 10.1109/ACCESS.2021.3077116.

[29] I. Siqueira, G. Correa, and M. Grellert, "Complexity and coding efficiency assessment of the versatile video coding standard," in *Proc. IEEE Int. Symp. Circuits Syst. (ISCAS)*, Daegu, South Korea, 2021, pp. 1–5, doi: 10.1109/ISCAS51556.2021.9401714.

[30] A. Cerveira, L. Agos tini, B. Zatt, and F. Sampaio, "Memory assessment of versatile video coding," in *Proc. IEEE Int. Conf. Image Process. (ICIP)*, Abu Dhabi, UAE, 2020, pp. 1186–1190, doi: 10.1109/ICIP40778.2020.9191358.

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

[32] W.-J. Chen *et al.*, *JVET AHG Report: Tool Reporting Procedure (AHG13)*, document JVET-S0013 of ITU-T/ISO/IEC Joint Video Experts Team (JVET), JVET, Jun./Jul. 2020.

Adam Wieckowski received the M.Sc. degree in computer engineering from the Technical University of Berlin, Berlin, Germany, in 2014.

In 2016, he joined the Fraunhofer Institute for Telecommunications, Heinrich Hertz Institute, Berlin, as a Research Assistant. He coordinated the development of HHIs' VVC proposal software, which later became the basis for VVC software test model VTM. He contributed several technical contributions during the standardization of VVC. Since 2019, he has been a Project Manager coordinating the technical development of decoder and encoder solutions for the VVC standard, VVdeC, and VVenC. His research interests include video coding, information theory, complexity reduction, and computer science.

Jens Brandenburg received the Dipl.-Ing. degree in computer engineering from the Technical University of Berlin, Germany, in 2004, and the Dr.-Ing. degree from the Friedrich–Alexander University Erlangen–Nürnberg, Germany, in 2018. He joined the Video Communication and Applications Department, Fraunhofer Institute for Telecommunications, Heinrich Hertz Institute, Berlin, in 2006, where he has been working on the implementation of different video decoder and encoder solutions for the H.264/AVC/SVC, H.265/HEVC, and the now emerging H.266/VVC standards. His research interests include HW/SW co-design for embedded systems and the optimization of video processing algorithms for real-time processing.

Benjamin Bross (Member, IEEE) received the Dipl.-Ing. degree in electrical engineering from RWTH Aachen University, Aachen, Germany, in 2008.

In 2009, he joined the Fraunhofer Institute for Telecommunications, Heinrich Hertz Institute, Berlin, Germany, where he is currently heading the Video Coding Systems Group, Video Coding and Applications Department and in 2011, he became a part-time Lecturer with the HTW Berlin University of Applied Sciences. Since 2010, he is very actively involved in the ITU-T VCEG | ISO/IEC MPEG video coding standardization processes as a Technical Contributor, the Coordinator of core experiments, and the Chief Editor of the High Efficiency Video Coding (HEVC) standard [ITU-T H.265 | ISO/IEC 23008-2] and the new Versatile Video Coding (VVC) standard [ITU-T H.266 | ISO/IEC 23090-3]. In addition to his involvement in standardization, his group is developing standard-compliant software implementations. This includes the development of an HEVC live software encoder that is currently deployed in broadcast for HD and UHD TV channels and most recently, the open and optimized VVC software implementations VVenC and VVdeC. He is an author or coauthor of several fundamental HEVC and VVC-related publications, and an author of two book chapters on HEVC and Inter-Picture Prediction Techniques in HEVC.

Mr. Bross received the IEEE Best Paper Award at the 2013 IEEE International Conference on Consumer Electronics, Berlin, in 2013, the SMPTE Journal Certificate of Merit in 2014, and an Emmy Award at the 69th Engineering Emmy Awards in 2017 as part of the Joint Collaborative Team on Video Coding for its development of HEVC.

Detlev Marpe (Fellow, IEEE) received the Dipl.-Math. degree (Highest Hons.) from the Technical University of Berlin, Berlin, Germany, in 1990, and the Dr.-Ing. degree from the University of Rostock, Rostock, Germany, in 2004.

He joined the Fraunhofer Institute for Telecommunications, Heinrich Hertz Institute, Berlin, in 1999, where he has been heading the Video Communication and Applications Department since 2015. He has successfully contributed to the three generations of modern video coding standards H.264/AVC, H.265/HEVC, and H.266/VVC, including most of their major enhancement extensions. He has authored numerous publications in the research area of image and video coding, and holds several hundreds of internationally issued patents in this area. His current research interests include still image and video coding, signal processing for communications and computer vision, machine learning, and information theory.

Dr. Marpe was a co-recipient of three Technical Emmy Awards as a Key Contributor and the Co-Editor of the H.264/MPEG-4 AVC Standard in 2008 and 2009, and as a Key Contributor of H.265/MPEG-HEVC in 2017, respectively. He received several best paper awards for his publications and he was recipient of the Karl Heinz Beckurts Award in 2011 and the Joseph von Fraunhofer Prize in 2004. From 2014 to 2018, he served as an Associate Editor for IEEE TRANSACTIONS ON CIRCUITS AND SYSTEMS FOR VIDEO TECHNOLOGY. In recognition of his dedicated contributions and excellent management of the review process, he was awarded the 2016 Best Associate Editor Award of the IEEE Circuits and Systems Society. He is a member of the Informationstechnische Gesellschaft of the Verband der Elektrotechnik Elektronik Informationstechnik e.V.