REAL-TIME IMPLEMENTATION OF SCALABLE HEVC ENCODER

Jaakko Laitinen, Ari Lemmetti, Jarno Vanne

Ultra Video Group, Tampere University, Finland
{jaakko.laitinen, ari.lemmetti, jarno.vanne}@tuni.fi

ABSTRACT

This paper presents the first known open-source Scalable HEVC (SHVC) encoder for real-time applications. Our proposal is built on top of Kvazaar HEVC encoder by extending its functionality with spatial and signal-to-noise ratio (SNR) scalable coding schemes. These two scalability schemes have been optimized for real-time coding by means of three parallelization techniques: 1) wavefront parallel processing (WPP); 2) overlapped wavefront (OWF); and 3) AVX2-optimized upsampling. On an 8-core Xeon W-2145 processor, the proposed spatially scalable Kvazaar can encode two-layer 1080p video above 50 fps with scaling ratios of 1.5 and 2. The respective coding gains are 18.4% and 9.9% over Kvazaar simulcast coding at similar speed. Correspondingly, the coding speed of SNR scalable Kvazaar exceeds 30 fps with two-layer 1080p video. On average, it obtains 1.20 times speedup and 17.0% better coding efficiency over the simulcast case. These results justify the benefits of the proposed scalability schemes in real-time SHVC coding.

Index Terms**—** High Efficiency Video Coding (HEVC), Scalable HEVC (SHVC), Kvazaar HEVC encoder, spatial scalability, signal-to-noise ratio (SNR) scalability

1. INTRODUCTION

The newest international video coding standard, *High Efficiency Video Coding* (*HEVC/H.265*) [1], [2], is developed to address increasing transmission and storage requirements of the latest video applications. It is published as a twin text by ITU, ISO, and IEC as ITU-T H.265 | ISO/IEC 23008-2.

The second version of the HEVC standard specifies a scalability extension called *Scalable HEVC* (*SHVC*) [3]. It enables video coding in multiple layers of different qualities. The SHVC bit stream contains only high-level syntax changes to the single-layer HEVC stream. The *base layer* (*BL*) of the stream represents the lowest quality. It is decodable with a standard HEVC decoder. Higher layers, referred to as *enhancement layers* (*ELs*), improve video quality and they can use lower ELs and the BL as *reference layers* (*RLs*).

SHVC addresses applications such as video streaming [4], broadcasting [5], and conferencing, where mobile devices may not be able to decode video at the highest quality. In these cases, media-aware network elements can reduce bandwidth requirements by only transmitting the BL. SHVC also provides the means to improve error resiliency. The BL can be sent over a reliable, low throughput channel and the EL over a less reliable, high throughput channel [6].

In practice, SHVC defines coding tools for spatial, *signal-to-noise ratio* (*SNR*), bit depth, color gamut, and hybrid codec scalability [3]. With spatial and SNR scalability, each layer represents a different video resolution and quality, respectively. Correspondingly, bit depth and color gamut scalability enable different bit depths and color spaces for each layer. Hybrid codec scalability, in turn, maintains backwards compatibility with previous standards by allowing for a non-HEVC BL.

SHVC introduces RLs to the EL inter prediction process and is thereby able to obtain bit rate savings over HEVC simulcast coding, where independent coding tasks are performed with corresponding quality and resolution parameters. However, different layer parameters (resolution, color space, etc.) need to be handled by a separate inter-layer processing step.

This paper presents a practical SHVC encoder implementation that conforms to the HEVC scalability extension. It is built on top of Kvazaar open-source HEVC encoder [7] and the latest version of its source code and issue tracker can be found on GitHub (https://github.com/ultravideo/scalable-kvazaar) under the GNU LGPLv2.1 license [8].

As far as we are aware, our proposal is the first real-time open-source SHVC encoder for spatial and SNR scalable coding. SHVC reference software called *SHVC Test Model* (*SHM*) [9] implements all normative SHVC coding tools for the best possible coding efficiency. However, huge computational complexity restricts its usage to research and conformance testing rather than practical encoding. R. Parois et al. [10] have presented a real-time SHVC encoder but it is based on a proprietary solution. Conversely, the other noteworthy open-source HEVC encoders, x265 [11], Turing codec [12], and SVT-HEVC Encoder [13], do not support scalability. On the other hand, there is one real-time SHVC decoder currently available [14].

The remainder of this paper is organized as follows. Section 2 presents an overview of our proposal, including the selected coding parameters of Kvazaar in Section 2.1, the proposed schemes for spatial and SNR scalable coding in Section 2.2, and finally, the parallelization techniques in Section 2.3. Section 3 reports performance results of the proposed schemes and compares them with Kvazaar simulcast coding and corresponding SHM results. Section 4 concludes the paper.

2. PROPOSED SCALABLE CODING SCHEMES

The proposed Scalable Kvazaar encoder is implemented by extending the Kvazaar ultrafast preset [7] with the coding tools for spatial and SNR scalability. Furthermore, all these scalable coding tools have been optimized to minimize their effect on coding speed.

2.1. Kvazaar ultrafast preset

Table I lists the main features of Kvazaar ultrafast preset. It supports HEVC Main profile with 8-bit YUV420 progressive input video.

Kvazaar encodes pictures in *coding tree units* (*CTUs*) of 64×64 luma pixels. Each CTU can be recursively divided into four equal-sized square *coding units* (*CUs*), and the division of the HEVC quadtree can be explored with depth-first search until the CUs of size 8×8 are reached.

In HEVC, CUs can be further split into rectangular-shaped *Prediction Units* (*PUs*) that share the identical prediction information. The Kvazaar ultrafast preset limits the search to the PUs of sizes 16×16 and 8×8. In inter prediction, *Hexagon-based search* (*HEXBS*) [15] is used in *motion estimation* (*ME*) with the *sum of absolute differences* (*SAD*) as the search criterion for the best *motion vectors* (*MVs*). Only one reference picture is used, and bi-directional ME is disabled by default. The minimum cost PU found by HEXBS is compared with existing merge candidates to see if a matching merge or skip mode can be used.

After inter prediction, Kvazaar performs the intra search using the *sum of absolute transformed differences* (*SATD*) as a distortion metric [16]. The prediction mode is finally selected by comparing the SATD costs of inter and intra predicted blocks. To speed up the mode decision, intra modes are not searched unless the previously obtained inter coding cost exceeds a predetermined threshold.

The prediction mode with the lowest cost is applied to reconstruct the CU and the associated prediction residual is transformed with integer *discrete cosine transform* (*DCT*) and quantized. The cost of a CU is estimated by computing the *sum of squared differences* (*SSD*) and the *context-adaptive binary arithmetic coding* (*CABAC*) cost, i.e., the number of bits needed for coding the prediction information and transform coefficients. The final partition of a CTU is composed of CUs with minimum coding costs.

2.2. Proposed scalability schemes for Scalable Kvazaar

SHVC only introduces high-level syntax changes to the original single-layer HEVC. Therefore, Scalable Kvazaar seeks to reuse the coding framework of the single-layer Kvazaar as much as possible and thereby minimize the overhead of the scalability extension. It has the same *application programming interface* (*API*) as the single-layer Kvazaar library in order to maintain compatibility with other applications. The largest differences can be found in the parameter set syntax (additional fields) and in the high-level structure, which has been modified to accommodate multiple layers.

Fig. 1 depicts a simplified block diagram of Scalable Kvazaar. It is made up of two Kvazaar encoder instances, one for the BL and the other for the EL, each with their layer specific parameters. These tightly coupled BL and EL encoder instances operate in parallel.

The spatial scalability scheme accepts input videos in the EL resolution and downsamples it to the BL resolution on the fly. Alternatively, the encoder can take two input streams if downsampling is performed beforehand. In the SNR (quality) scalability scheme, the EL encoder is assigned a smaller *quantization parameter* (*QP*) than the BL encoder so that the EL is encoded with a higher quality than the BL.

In Scalable Kvazaar, the BL and EL encoders encode the input video with the respective layer parameters. The BL encoder only uses the previous BL picture as a reference for inter prediction, whereas the EL encoder utilizes both the previous EL picture and an *inter-layer reference* (*ILR*) picture – a BL picture with the same *picture order count* (*POC*). The ILR picture is added when updating the reference picture list in the EL encoder. Moreover, the SHVC standard specifies that the ILR picture needs to be marked as a long-term reference picture.

SHVC specifies both texture and motion prediction tools [3] that can be used with ILR pictures. Texture prediction is similar to normal inter prediction, but ME is completely omitted by directly setting all MVs to zero, i.e., a collocated CU in the BL is always used as a reference. When *temporal MV prediction* (*TMVP*) is enabled, motion prediction is conducted and the ILR picture is set as the collocated TMVP reference picture in the EL encoder. In this case, the ILR picture is used to derive a temporal MV candidate for TMVP.

Table I. Coding features of Kvazaar ultrafast preset.

Feature	Kvazaar parametrization
Input format	8-bit YUV420
Coding units	64×64, 32×32, 16×16, 8×8
Inter prediction units	16×16, 8×8
Motion estimation algorithm	HEXBS
Reference pictures	1
Temporal MV prediction	Enabled
Fractional motion estimation	Enabled
Bi-prediction	Disabled
Intra prediction units	64×64, 32×32, 16×16, 8×8
Intra prediction modes	35 (DC, planar, 33 angular)
Mode decision metrics	SAD, SATD, SSD, CABAC
Transform units	32×32, 16×16, 8×8
Loop filters	Deblocking
Parallelization	WPP, OWF

There is a one-to-one correspondence between the spatial resolutions of the BL and EL in the SNR scalability scheme, so the EL encoder can use BL texture and motion information directly as is. However, the spatial scalability scheme requires an additional inter-layer processing step to upsample BL texture and motion information of the ILR picture for the EL encoder. The texture upsampling process is carried out with interpolation filters defined in the SHVC standard [1]. These filters allow for practically arbitrary scaling ratios between the BL and EL resolutions. The motion field of the ILR picture is generated in 16×16 blocks by propagating the motion information from the collocated BL block and scaling up the MVs according to the scaling ratios between the BL and EL. The block size of 16×16 is used for motion information in order to decrease the memory footprint [3].

The encoding process is completed by concatenating the BL and EL bitstreams to form the final SHVC conformant bitstream. The bitstream will be a mix of *network abstraction layer* (*NAL*) units from the BL and EL. They are distinguished by the layer ID – zero for BL and one for EL – in the NAL unit header.

2.3. Parallelization

In this work, Scalable Kvazaar is designed to take advantage of the existing thread- and data-level parallelization tools of the single-layer Kvazaar. Furthermore, the additional inter-layer processing step has been optimized and parallelized accordingly.

At the thread level, Scalable Kvazaar supports *wavefront parallel processing* (*WPP*) [17] and o*verlapped wavefront* (*OWF*) [18] parallelization strategies simultaneously [19]. In addition, the upsampling processes for texture and motion information have been customized for WPP by dividing them into parallel tasks that upsample one BL CTU at a time. The control logic of WPP is synchronized to guarantee that the required BL texture and motion information is available before upsampling begins. Correspondingly, the EL encoder waits for the completion of the needed upsampling results.

OWF is a picture-level parallel processing tool that allows multiple pictures to be encoded simultaneously. This is beneficial

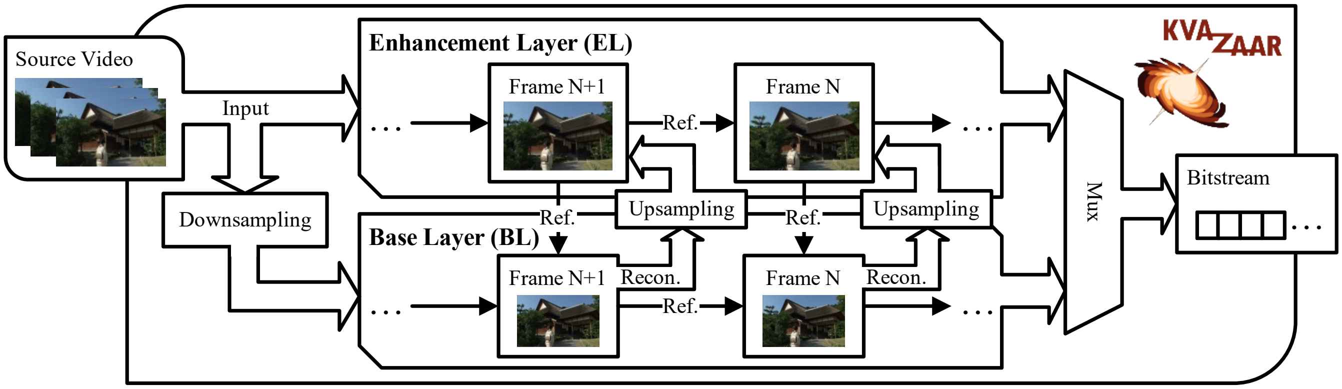

Fig. 1. Simplified block diagram of the proposed Scalable Kvazaar encoder.

because there tends to be more tasks available for execution even though dependencies limit the execution of tasks in a single picture. It also improves CPU utilization by hiding the overhead of starting to encode a new picture. By dividing the upsampling into smaller tasks and integrating it with WPP, the benefits of OWF are unlocked automatically.

At the data level, both the BL and EL encoders have been made compatible with the previously implemented *Advanced Vector Extensions 2* (*AVX2*) – a *single instruction multiple data* (*SIMD*) processor instruction set – optimizations [20]. In Scalable Kvazaar, texture upsampling has been AVX2-optimized for maximal throughput. In the first pass, the 8-bit input values are loaded from the given texture block into 256-bit registers for the horizontal upsampling step. Using '_mm256_maddubs_epi16' and addition intrinsic functions, sixteen pixels can be processed with 8-tap filters in one batch. The resulting 16-bit intermediate values are stored for the vertical pass, where they are passed together with filter coefficients to '_mm256_madd_epi16' and addition intrinsic functions that calculate the final values for eight pixels simultaneously. The final values are scaled back to 8-bit upsampled texture values. The scaling can be done to 8 pixels at a time to make the most of the 256-bit registers.

3. PERFORMANCE EVALUATION

In our experiments, two-layer (BL + EL) test video sequences were used to benchmark the coding efficiency and speed of Scalable Kvazaar v1.0.0 against the corresponding Kvazaar simulcast coding. The respective tests were also conducted with SHM12.1 and *HEVC reference encoder* (*HM16.10*) [21] for the sake of comparison.

Table II lists the parameters used to run the simulcast (BL and EL) and spatial/SNR scalability tests under a low-delay P coding configuration with intra frame period of 64. The simulcast results were obtained by aggregating the BL and EL encoding results of single-layer Kvazaar. For SHM and HM, the low-delay P configuration and sequence-specific configuration files were used and the QP values were set manually on the command line.

Altogether, seven full-length 8-bit YUV420 video sequences were taken from the common SHM test conditions [22]. For the spatial scalability comparisons, the input sequences were pre-scaled from the EL resolution to that of the BL according to the used scaling ratios of 2× or 1.5×.

The test platform was an 8-core Intel® Xeon ® W-2145 CPU with 32 GB of RAM (DDR4 2666 ECC), running 64-bit Microsoft Windows 10. The processor supports SIMD extensions up to AVX2.

Table II. Parameters of simulcast and Scalable Kvazaar.

Encoder	Parameters
Simulcast Kvazaar	--input=<layer sequence> --preset=ultrafast --threads=15 --owf=2 -q <layer qp>
Scalable Kvazaar	--input=<bl sequence> --preset=ultrafast --threads=15 --owf=2 -q <bl qp> --layer --input=<el sequence> --preset=ultrafast --threads=15 --owf=2 -q <el qp>

3.1. Coding efficiency

Table III provides the bit rate gain of spatially scalable Kvazaar (2× and 1.5×) over single-layer Kvazaar, which encodes the BL and EL sequentially. Both the total bit rate (EL+BL) savings and EL-only savings are reported for all test cases, except for the 1600p resolution that is not evenly divisible by 1.5×.

Bjøntegaard delta bitrate (*BD-rate*) [23] with piecewise cubic interpolation [24] was used in the bit rate comparison. The default QP values used to calculate the BD-rates were 22, 27, 32, and 37.

In summary, an average BD-rate gain of 9.9% is achieved with a scaling ratio of 2× and 18.4% improvement with the ratio of 1.5×. The respective savings for the EL-only case are 10.9% and 26.5%. Even with the loss introduced by upsampling, the EL encoder is able to take advantage of the texture and motion information obtained from the BL.

Table IV tabulates the corresponding results for SNR scalable Kvazaar with two different *delta QP* (Δ*QP*) values between the EL and BL, where the aforementioned default QP values are used in the BL. SNR scalable Kvazaar improves BD-rate over the simulcast case by 8.3% with ΔQP = -9 or 9.8% when comparing the ELs only. The respective results are 25.7% or 41.9% for ΔQP = -3. The higher the delta QP value, the harder it is to find blocks with low coding cost in the lower quality BL. The qualities of the BL and EL converge with smaller delta QP values, which allows the EL encoder to use blocks from the BL more efficiently and thereby obtain higher BD-rate savings.

Tables III and IV also include BD-rate results between SHM and HM simulcast coding. In all test cases, the relative coding gains of SHM are fairly similar to that of Scalable Kvazaar. For spatial scalability, SHM attains slightly higher bit rate savings (10.5% and 18.6%), whereas for SNR scalability, SHM gives smaller bit rate gains (6.9% and 24.3%).

Table III. Coding gain and speedup of spatially Scalable Kvazaar and SHM over Kvazaar and HM simulcast coding, respectively.

		Scalable Kvazaar								SHM			
		2×				1.5×				2x		1.5x	
		BD-rate				BD-rate				BD-rate		BD-rate	
Format	Sequence	EL+BL	EL	Δspeed	Speed	EL+BL	EL	Δspeed	Speed	EL+BL	Δspeed	EL+BL	Δspeed
2560×1600 (1600p)	PeopleOnStreet	-14.5%	-19.9%	0.99×	24 fps	-	-	-	-	-14.9%	1.26×	-	-
	Traffic	-3.3%	-1.1%	0.89×	34 fps	-	-	-	-	-8.2%	1.20×	-	-
1920×1080 (1080p)	BQTerrace	-4.6%	-3.4%	0.98×	55 fps	-9.8%	-12.3%	1.00×	51 fps	-8.2%	1.18×	-12.1%	0.94×
	BasketballDrive	-13.2%	-14.1%	1.06×	57 fps	-22.6%	-32.5%	1.11×	54 fps	-11.3%	1.24×	-22.8%	1.01×
	Cactus	-9.2%	-7.7%	1.01×	61 fps	-18.9%	-26.4%	1.07×	56 fps	-9.8%	1.21×	-18.6%	0.98×
	Kimono	-15.7%	-19.8%	1.00×	61 fps	-24.6%	-37.5%	1.08×	58 fps	-13.4%	1.24×	-25.4%	1.02×
	ParkScene	-9.1%	-10.3%	0.98×	58 fps	-16.4%	-24.0%	1.02×	54 fps	-7.6%	1.12×	-14.3%	0.99×
	Average	-9.9%	-10.9%	0.99×		-18.4%	-26.5%	1.05×		-10.5%	1.21×	-18.6%	0.99×

Table IV. Coding gain and speedup of SNR Scalable Kvazaar and SHM over Kvazaar and HM simulcast coding, respectively.

		Scalable Kvazaar								SHM			
		ΔQP = -9				ΔQP = -3				ΔQP = -9		ΔQP = -3	
		BD-rate				BD-rate				BD-rate		BD-rate	
Format	Sequence	EL+BL	EL	Δspeed	Speed	EL+BL	EL	Δspeed	Speed	EL+BL	Δspeed	EL+BL	Δspeed
2560×1600 (1600p)	PeopleOnStreet	-9.3%	-11.1%	1.11×	16 fps	-33.2%	-57.4%	1.26×	20 fps	-9.1%	1.04×	-32.5%	1.10×
	Traffic	-2.1%	-0.9%	1.10×	23 fps	-17.7%	-28.3%	1.18×	28 fps	-5.4%	1.02×	-17.6%	1.03×
1920×1080 (1080p)	BQTerrace	-8.6%	-9.6%	1.16×	32 fps	-21.7%	-31.9%	1.23×	41 fps	-6.4%	1.05×	-21.5%	1.07×
	BasketballDrive	-10.7%	-13.1%	1.16×	35 fps	-31.0%	-51.1%	1.30×	45 fps	-8.4%	1.06×	-28.5%	1.09×
	Cactus	-10.7%	-12.8%	1.18×	37 fps	-25.5%	-40.1%	1.26×	47 fps	-6.9%	1.06×	-22.5%	1.09×
	Kimono	-10.9%	-14.4%	1.16×	39 fps	-29.2%	-50.0%	1.30×	51 fps	-7.1%	1.06×	-28.1%	1.12×
	ParkScene	-6.0%	-7.0%	1.15×	38 fps	-21.4%	-34.3%	1.24×	48 fps	-5.2%	1.01×	-19.3%	1.03×
	Average	-8.3%	-9.8%	1.14×		-25.7%	-41.9%	1.25×		-6.9%	1.04×	-24.3%	1.08×

3.2. Coding speed

Scalable Kvazaar is capable of taking advantage of all optimized and parallelized coding tools of Kvazaar, which has been shown to provide a notable speedup over HM [25]. Furthermore, SNR scalability does not require any additional inter-layer processing steps; only the number of reference pictures increases, by one, when adding an ILR picture. As stated above, this is inexpensive, since the SHVC standard specifies that the MVs for ILRs are always set to zero [1]. Thus, there is no need to perform any ME on ILR pictures.

Table III reports the speedup of spatially scalable Kvazaar over the simulcast case. The absolute speed of scalable encoding is also given in *frames per second* (*fps*), as an average over the given four QP values. With 2× scalability, Scalable Kvazaar is slightly slower on average than Kvazaar simulcast coding due to the upsampling overhead. In this case, the speed gain obtained with simultaneous BL and EL processing is inherently limited because the EL encoder dominates the processing time. With the scaling ratio of 1.5×, however, the processing times of the BL and EL encoders are better balanced for simultaneous processing and Scalable Kvazaar ends up achieving a 1.05× speedup on the 1080p sequences. The spatial scalability scheme of Scalable Kvazaar is, on average, able to reach a real-time coding speed of 57 fps for 1080p.

Table IV tabulates the respective coding speeds for the SNR scalability case. The BL and EL encoders work on the same resolution, which balances their processing times and further mitigates the inherent limitations of WPP at the frame boundaries. As a result, Scalable Kvazaar is, on average, 1.20× as fast as Kvazaar simulcast coding. When all optimizations are enabled, Scalable Kvazaar is able to reach coding speeds of over 32 fps for 1080p sequences. Hence, it is practical for real-time applications of up to 1080p resolution on the selected platform.

Tables III and IV also report the speed differences between SHM and HM simulcast coding in the examined cases. The absolute speed results are omitted because they are in the order of minutes per frame, i.e., far from real-time encoding. Nevertheless, the speedup of SHM is at the same level as that of Scalable Kvazaar.

4. CONCLUSIONS

This paper presented the first practical open-source SHVC encoder called Scalable Kvazaar. It is designed for spatial and SNR scalable encoding. The reported results show that the proposed spatial and SNR scalability schemes are able to reach average BD-rate savings of 9.9% - 18.4% and 8.3% - 25.7% over the corresponding Kvazaar simulcast coding, respectively. The carefully parallelized inter-layer processing and AVX2-optimized texture upsampling make it possible to obtain the reported coding gains with negligible effect on coding speed in the spatial scalability case and a 1.20× speedup in the SNR scalability case. On an 8-core Xeon® processor, Scalable Kvazaar attains a coding speed of over 30 fps for 1080p two-layer video, making it practical for real-time SHVC applications.

ACKNOWLEDGMENTS

This work was supported in part by the European Celtic-Plus Project Virtuose and the Academy of Finland (decision no. 301820). The authors would also like to thank all contributors of Kvazaar open-source project [7].

6. REFERENCES

[1] High Efficiency Video Coding, *document ITU-T Rec. H.265 and ISO/IEC 23008-2 (HEVC)*, ITU-T and ISO/IEC, Nov. 2019.

[2] G. J. Sullivan, J. R. Ohm, W. J. Han, and T. Wiegand, “Overview of the High Efficiency Video Coding (HEVC) standard,” *IEEE Trans. Circuits Syst. Video Technol*., vol. 22, no. 12, Dec. 2012, pp. 1649-1668.

[3] J. M. Boyce, Y. Ye, J. Chen, and A. K. Ramasubramonian, “Overview of SHVC: Scalable extension of the High Efficiency Video Coding standard,” *IEEE Trans. Circuits Syst. Video Technol*., vol. 26, no. 1, Jan. 2016, pp. 20-34.

[4] Y. Ye, Y. He, and X. Xiu, “Manipulating ultra-high definition video traffic,” *IEEE MultiMedia*, vol. 22, no. 3, July-Sept. 2015, pp. 73-81.

[5] R. Parois, W. Hamidouche, P. Cabarat, M. Raulet, N. Sidaty, and O. Déforges, “4K real time software solution of scalable HEVC for broadcast video application,” *IEEE Access*, vol. 7, Mar. 2019, pp. 46748-46762.

[6] P. Ronan, T. Eric, and R. Mickaël, “Hybrid broadband/broadcast ATSC 3.0 SHVC distribution chain,” in *Proc. IEEE Int. Symp. on Broadband Multimedia Syst. and Broadcasting*, Valencia, Spain, June 2018.

[7] Ultra Video Group. *Kvazaar open-source HEVC Encoder*. [Online]. Available: https://github.com/ultravideo/kvazaar.

[8] Ultra Video Group. *Scalable Kvazaar*. [Online]. Available: https://github.com/ultravideo/scalable-kvazaar.

[9] JCT-VC. *SHVC Reference Software, ver. SHM 12.1*. [Online]. Available: http://hevc.hhi.fraunhofer.de/shvc.

[10] R. Parois, W. Hamidouche, J. Vieron, M. Raulet, and O. Déforges, “Efficient parallel architecture for a real-time UHD scalable HEVC encoder,” in *Proc. European Signal Process. Conf*., Kos, Greece, Aug.-Sept. 2017.

[11] MulticoreWare Inc. *x265 HEVC Encoder.* [Online]. Available: https://bitbucket.org/multicoreware/x265/downloads/.

[12] Turingcodec.org. *Turing codec.* [Online]. Available: http://turingcodec.org/.

[13] 01.org. *Scalable Video Technology for HEVC Encoder (SVT-HEVC Encoder).* [Online]. Available: https://github.com/OpenVisualCloud/SVT-HEVC

[14] W. Hamidouche, M. Raulet, and O. Déforges, “4K real-time and parallel software video decoder for multilayer HEVC extensions,” *IEEE Trans. Circuits Syst. Video Technol*., vol. 26, no. 1, Jan. 2016, pp. 169-180.

[15] C. Zhu, X. Lin, and L. P. Chau, “Hexagon-based search pattern for fast block motion estimation,” *IEEE Trans. Circuits Syst. Video Technol*., vol. 12, no. 5, May 2002, pp. 349-355.

[16] M. Viitanen, A. Koivula, A. Lemmetti, J. Vanne, and T. D. Hämäläinen, “Kvazaar HEVC encoder for efficient intra coding,” in *Proc. IEEE Int. Symp. Circuits Syst*., Lisbon, Portugal, May 2015, pp. 1662-1665.

[17] G. Clare, F Henry, and S. Pateux, “Wavefront parallel processing for HEVC encoding and decoding,” *document JCTVC-F274*, Torino, Italy, July 2011.

[18] C. C. Chi, M. Alvarez-Mesa, B. Juurlink, V. George, and T. Schierl, “Improving the parallelization efficiency of HEVC decoding,” in *Proc. IEEE Int. Conf. Image Processing*, Orlando, Florida, USA, Sept.-Oct. 2012.

[19] A. Koivula, M. Viitanen, J. Vanne, T. D. Hämäläinen, and L. Fasnacht, “Parallelization of Kvazaar HEVC intra encoder for multi-core processors,” in *Proc. IEEE Workshop Signal Process. Syst*., Hangzhou, China, Oct. 2015, pp. 1-6.

[20] A. Lemmetti, A. Koivula, M. Viitanen, J. Vanne, and T. D. Hämäläinen, “AVX2-optimized Kvazaar HEVC intra encoder,” in *Proc. IEEE Int. Conf. Image Processing*, Phoenix, Arizona, USA, Sept. 2016.

[21] JCT-VC. *HEVC Reference Software, ver. HM 16.10*. [Online]. Available: http://hevc.hhi.fraunhofer.de/.

[22] V. Sergin and Y. He, “Common SHM test conditions and software reference configurations,” *document JCTV-Q1009*, Valencia, Spain, Mar.-Apr. 2014.

[23] G. Bjøntegaard, “Calculation of average PSNR differences between RD curves,” *document VCEG-M33*, Austin, Texas, USA, pp. 1-4, Apr. 2001.

[24] J. Wang, X. Yu, and D. He, “On BD-rate calculation,” *document JCTV-F270*, Torino, Italy, July 2011.

[25] A. Lemmetti, M. Viitanen, A. Mercat, and J. Vanne, “Kvazaar 2.0: fast and efficient open-source HEVC inter encoder,” in *Proc. ACM Multimedia Syst. Conf.*, Istanbul, Turkey, June 2020.