TECHNICAL PAPER

High-Throughput JPEG 2000 for Broadcast and IP-Based Applications

By Thomas Edwards and Michael D. Smith

Abstract

High-Throughput JPEG 2000 (HTJ2K) is a royalty-free image compression standard published in 2019 that enhances JPEG 2000 by replacing its slow block coder with a fast block coder. The resulting speedup (e.g., >30× for lossless coding) accelerates the encoding and decoding of images, which can have a great impact on users. A less obvious benefit of the speedup is the enabling of low-latency applications, which was not possible with the original JPEG 2000 standard. Examples of these new potential applications are live, high-quality, low-latency broadcast video contribution, remote production, and Internet Protocol (IP)-based production on-premises and on cloud. This paper examines various HTJ2K encoding parameters and their impact on quality, bitrate, latency, and multigeneration encode/decode cycle performance in the context of requirements of broadcast and IP-based applications. Configurations include a number of different wavelet filters, code-block sizes, and wavelet decomposition structures that are available with HTJ2K. The performance of low-latency HTJ2K is also compared to other low-latency wavelet codecs like VC-2, JPEG XS, and JPEG 2000 Part-1. The JPEG 2000 Part-1 comparisons use the full-frame broadcast profile as well as the ultra-low latency (ULL) configuration as per Video Services Forum (VSF) Technical Recommendation (TR)-01:2018. A tradeoff between latency and compression quality is discussed.

Keywords

Contribution, High-Throughput JPEG 2000 (HTJ2K), internet protocol (IP) production, JPEG 2000, JPEG XS, low latency, VC-2

Digital Object Identifier 10.5594/JMI.2021.3066183
Date of publication: 5 May 2021

Introduction

"There ain't no such thing as a free lunch," is a popular adage. Also, in the transmission of media at a constant bitrate, there is an inherent tradeoff between quality, computational energy, and latency. In use cases, such as remote production and cloud-based Internet Protocol (IP) production, the high data rates of uncompressed video motivate the investigation of media transmission that is quite low in both data rate and latency when it comes to live production. Cloud-based production may require the video to flow through many processing steps, emphasizing that good multigenerational performance of a codec is vital. Furthermore, cloud-based production is likely to require compression that is computationally efficient for software-based media-processing systems running on commodity central-processing units (CPUs), as opposed to only being practical in custom hardware such as field-programmable gate arrays (FPGAs).

"There ain't no such thing as a free lunch," is a popular adage. Also, in the transmission of media at a constant bitrate, there is an inherent tradeoff between quality, computational energy, and latency.

This paper explores a class of wavelet-based video codecs that promise subframe latency and efficient implementation in software while providing enough bitrate reduction from uncompressed to make the extra computation worthwhile. In particular, we present results of a new codec known as *High-Throughput JPEG 2000 (HTJ2K)* and its operations at several configurations with different amounts of subframe latency. Comparison is also made with full-frame JPEG 2000, which is commonly used for contribution and interfacility transmission, but generally considered too complex for realtime implementation on CPUs at high resolutions and data rates that correspond to very high image quality.

Low-Latency Codecs Under Test

In general, low latency can be achieved with wavelet-based codecs using either tile-based or precinct-based structures and rate control. Tile-based rate control is relatively simple; the input image is split into independent contiguous sections and each image section is compressed independently, targeting a specific number of bytes to meet a constant bitrate operating constraint. As the independent tiled sections get smaller, both the latency and the compression efficiency are reduced.

Precincts are spatial groupings of wavelet coefficients. Precinct-based rate control segments the image data into small units within the wavelet domain, which eliminates tile boundary artifacts and improves compression efficiency when moving to sub-100 line latencies.[1] When operating an image codec at very low latencies, high-level parallelism is difficult to exploit, which can make achieving high throughput a challenge unless the entropy coding is very simple, provides additional concurrency, and has few serial dependencies. The dependencies in the buffering model can also introduce challenges to achieving high throughput in software with multithreaded implementations.[2] Other precinct-based wavelet codecs that are available to the industry, but not evaluated in this paper, include SMPTE Registered Disclosure Document (RDD) 34 (LLVC—Low-Latency Video Codec for Network Transfer) and SMPTE RDD 35 (TICO Lightweight Codec Used in IP Networked or SDI Infrastructures).

Algorithmic dependency analysis can be performed to determine the lowest theoretical end-to-end latency, assuming the input video lines arrive in a raster-scan order sequentially in time. This analysis considers the region of support required for the concatenated wavelet transform-filtering structures, in addition to the grouping of compressed data elements and the buffering model. However, such analysis does not consider the time required to perform the actual computations. If the required computations cannot be completed in time to keep up with the source data rate, the solution will not work in practice. For example, such algorithmic dependency analysis could be performed with JPEG 2000 Part-1 using very short precincts and few vertical wavelet transforms and will show very low algorithmic latency, but once actual computation time is considered, a practical system would not be able to operate at high throughput to keep up with a high-definition (HD) resolution source, unless it was implemented on an application-specific integrated circuit (ASIC) running at very high clock rates (1 GHz or higher). Richter *et al.*[2] examined tradeoffs between latency, quality, and throughput for the JPEG XS codec implemented on a multithreaded software platform.

This paper investigates the coding performance of four wavelet-based codecs. JPEG 2000 Part-1 (J2K) is used in a full-frame mode and a tile-based stripe mode, both using 9/7 wavelet filters. HTJ2K is used with different configurations of a wavelet transform filter kernel (5/3 and 9/7) and different numbers of vertical wavelet transform levels (0–3) while always using five horizontal wavelet transform levels. JPEG XS is used in its configurations complying with main and high profiles using one and two vertical wavelet transforms, respectively, both using the 5/3 wavelet filter. VC-2 is evaluated with three-wavelet transform levels and the 5/3 wavelet filter kernel. The 5/3 filter set has fewer filter taps than the 9/7 filter set and can therefore lead to lower latencies, but generally has worse compression performance than when the same codec is used with the 9/7 filter set. The Haar wavelet has the shortest filters and therefore leads to the lowest latencies, but has even worse compression performance than the 5/3 filter.

JPEG 2000 (J2K) "Full-Frame"

JPEG 2000 Part 1 [Rec. International Telecommunications Union-Telecommunications (ITU-T) T.800| International Organization for Standardization/International Electrotechnical Commission (ISO/IEC) 15444-1] is a widely used codec for contribution and digital cinema that can deliver 10:1 compression with visual transparency even through multiple codec passes. J2K is based on a two-dimensional discrete wavelet transform (DWT), a uniform dead-zone quantizer, and an entropy coder based on embedded block coding with optimal truncation (EBCOT) in combination with an adaptive context-based arithmetic coder (MQ-coder). Unfortunately, its high computational complexity makes software implementation very challenging. JPEG 2000 is typically used to code an entire frame, which results in significant end-to-end latency, often three frames in shipping products implementing Video Services Forum (VSF) Technical Recommendation (TR)-01:2013.

JPEG 2000 Part 1 was developed by JPEG to be royalty-free, and the technologies used in the original standard are now close to 20 years old, which generally is the maximum length of patent protection. However, it should be noted that Annex L of ISO/IEC 15444-1:2019 states that compliance with the standard "may involve the use of patents" and that "the complete list of intellectual property rights statements that have been received can be obtained from the ITU-T and ISO patent declaration databases."

JPEG 2000 full-frame as tested in this paper conformed to the JPEG 2000 Broadcast Contribution Single Tile Profile with Level 5. This configuration uses the 9/7 irreversible wavelet filter with five horizontal and five vertical wavelet transform levels with 64 × 64 codeblocks. All J2K full-frame, J2K ultralow latency (ULL), and HTJ2K codec compression testing in this paper was performed with Kakadu Software.[3]

JPEG 2000 ULL (J2K ULL)

"JPEG 2000 Ultra-Low Latency" (defined as part of Rec. ITU-T H.222.0 via ISO/IEC 13818-1:2018 AMD1) is a method of transmitting J2K-coded images via the use of "stripes" comprising an independently decodable horizontally divided portion of an image to provide subframe latency. All stripes in J2K ULL have the same size, except for the very bottom stripe, which may have a different height. Latency becomes lower when the stripe height is small. Algorithmic latency is twice the stripe height.

One drawback of image-domain stripes is that a stripe boundary can be visible because image data are coded independently on either side of the stripe boundary. The terminology "stripe" is specific to J2K ULL. In more common JPEG 2000 terminology, stripes are conceptually similar to "tiles," while in Motion Picture Experts Group (MPEG) terminology, stripes are conceptually similar to "slices." More details on the use of J2K ULL over IP can be found in VSF TR-01:2018.[4] It is widely believed that the addition of the stripe signaling and carriage in J2K ULL implementations requires no royalty.

VSF TR-01:2018 defines several profiles for J2K ULL stripes, and within those profiles, a range of allowed stripes per frame and the number of decomposition levels. In profile "3" (named "3G") intended for use with 1080p at 50 and 59.94 frames/sec, between 4 and 9 stripes are allowed. For this test, two configurations were used: one with four stripes and four decomposition levels, and another one with nine stripes and three decomposition levels. Both used irreversible 9/7 wavelet decomposition with a codeblock size of 32×32.

Given the lack of native J2K ULL test software, FFMPEG was used to split the input uncompressed YUV frame sequences into YUV stripe frame sequences. Those stripe sequences were then compressed to JPEG 2000 and decompressed using Kakadu software, and the resulting output stripe sequences were recombined using FFMPEG before peak signal-to-noise ratio (PSNR) and structural similarity (SSIM) evaluation to the source reference.

High-Throughput JPEG 2000 (HTJ2K)

The "high throughput" or "HT" block coder (Rec. ITU-T T.814|ISO/IEC 15444-15) can replace the original J2K block coder to reduce computational complexity by an order of magnitude, at the expense of some loss in coding efficiency. The result is known as *High-Throughput JPEG 2000* or *HTJ2K*.

The details of the HTJ2K block coder are described elsewhere[1,5] but are summarized here in comparison to the original J2K block coder. Due to the design goal of facilitating lossless transcoding of J2K to/from HTJ2K, the HTJ2K block also uses a fractional bitplane segmentation using Cleanup, SigProp, and MagRef coding passes. Although J2K codes each coding pass serially, HTJ2K codes all the fractional bitplanes in one Cleanup pass except for the least significant bit (LSB) bitplane which can include SigProp and MagRef passes. In addition to the collapsing of the coding of bitplanes, HTJ2K also permits more concurrency in the entropy coding with fewer dependencies and shorter critical paths.

HTJ2K can be operated with precinct-based rate control using short precincts and a few vertical wavelet transforms to achieve ULLs while avoiding the potential of stripe-boundary artifacts that may be found using J2K ULL. Due to HTJ2K's low complexity and potential for concurrency within the entropy coder, it is possible to practically achieve both ULL and high throughput at low clock rates on FPGAs and with modern commodity CPUs, which is not possible with J2K. Because HTJ2K simply replaces the block coder in JPEG 2000 Part-1, other tools from JPEG 2000 Part-2 can also be combined with HTJ2K. Part-2's additional tools supporting arbitrary wavelet kernels and fewer vertical wavelet transforms than horizontal wavelet transforms can be used to offer additional capabilities and high performance for low-latency applications.

JPEG 2000 Part-1 supports two different wavelet kernel sets: the "irreversible 9/7" that uses irrational filter coefficients, and "reversible 5/3" that uses rational filter coefficients. Irreversible 9/7 is typically used for lossy coding, and reversible 5/3 is typically used for lossless coding. Using the arbitrary wavelet kernel feature of Part-2 allows using other wavelet filter sets, like the irreversible 5/3. Using the irreversible 5/3 generally results in worse compression performance compared to the irreversible 9/7 filters, but the filters are shorter and therefore can reduce latency in comparison. Another option would be to use reversible 5/3 which has the same latency as irreversible 5/3 but has worse compression performance due to the nonlinear rounding steps in reversible transforms, which changes the quantization errors. In this paper, we evaluate the performance and latency of HTJ2K configurations using both irreversible 5/3 and irreversible 9/7 filters. When using the irreversible 5/3 wavelet filters, the HTJ2K codestreams must include the arbitrary transform kernel (ATK) marker segment. The z-transform of the irreversible 5/3 analysis filter set is shown in the following equation:

$$\text{Analysis LPF: } -1/8\, z^2 + 1/4\, z + 3/4 + 1/4\, z^{-1} - 1/8\, z^{-2}$$

$$\text{Analysis HPF: } -1/4\, z^2 + 1/2\, z - 1/4.$$

JPEG 2000 Part-1 requires that the same number of wavelet transform levels are applied in the horizontal and vertical dimensions. JPEG 2000 Part-2 allows a different number of wavelet transforms to be applied in the horizontal and vertical dimensions, using the downsample factor style (DFS) and arbitrary decomposition style (ADS) marker segments. The upper-left diagram in **Fig. 1** shows a five-level symmetric horizontal/vertical wavelet

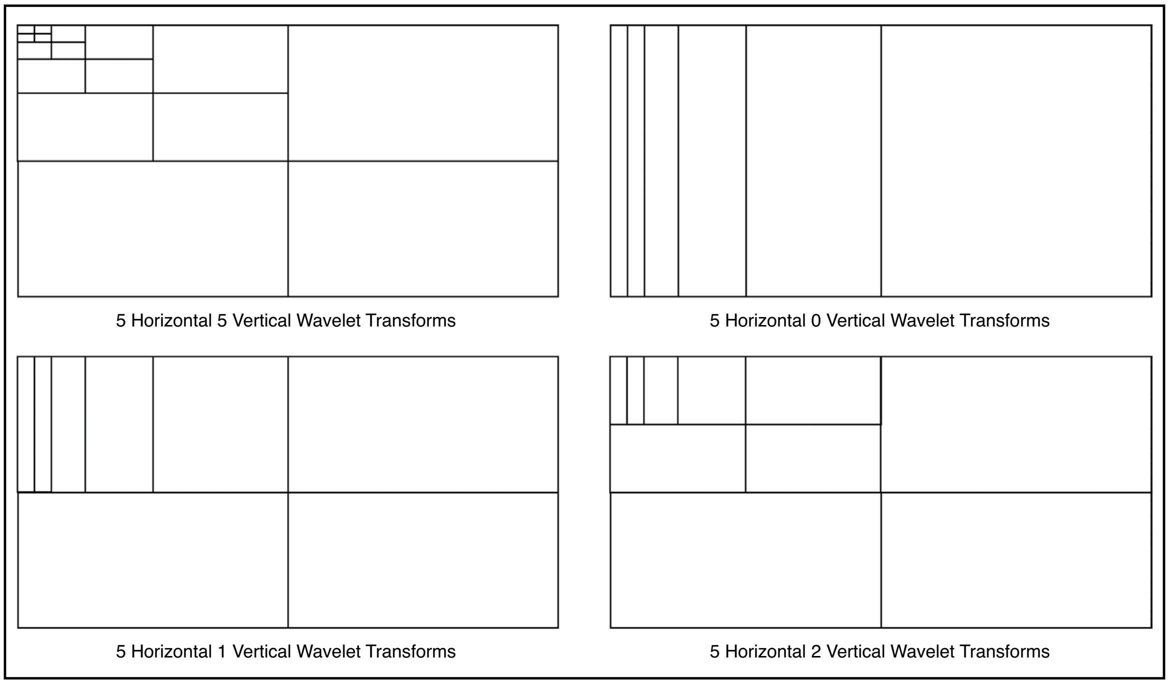
5 Horizontal 5 Vertical Wavelet Transforms
5 Horizontal 0 Vertical Wavelet Transforms
5 Horizontal 1 Vertical Wavelet Transforms
5 Horizontal 2 Vertical Wavelet Transforms

FIGURE 1. Diagram of wavelet transform decomposition structures.

transform. The other three diagrams show asymmetric decomposition structures, which are more optimized to low-latency applications, with 0–2 vertical wavelet transform levels and five horizontal wavelet transform levels.

Table 1 is a summary of the subframe latency HTJ2K configurations used for the testing presented in this paper. Note the use of the abbreviation "VWT" for the "number of vertical wavelet transform levels."

The configurations described above are all minimal latency configurations for the given number of vertical wavelet transforms except for the HTJ2K 9/7, a 3 VWT configuration, which uses taller precincts than the

TABLE 1. HTJ2K configurations in this paper.

HT configuration Label	HTJ2K 9/7, 0 VWT	HTJ2K 9/7, 1 VWT	HTJ2K 9/7, 2 VWT	HTJ2K 9/7, 3 VWT	HTJ2K 5/3, 1 VWT	HTJ2K 5/3, 2 VWT	HTJ2K 5/3, 3 VWT
Codeblock size (height, width)	(4, 1024)	(4, 1024)	(4, 1024)	(16, 256)	(4, 1024)	(4, 1024)	(8, 512)
Codeblock style	0 × 40						
Number of decomposition levels	5						
Progression order	PCRL (Position-Component-Resolution-Layer)						
Wavelet filter	9/7 irreversible				5/3 irreversible		
Vertical wavelet transforms	0	1	2	3	1	2	3
Precinct size R0 (height, width)	(2, 8192)	(2, 8192)	(2, 8192)	(8, 8192)	(2, 8192)	(2, 8192)	(2, 8192)
Precinct size R1 (height, width)	(2,8192)	(2,8192)	(2,8192)	(8,8192)	(2,8192)	(2,8192)	(2,8192)
Precinct size R2 (height, width)	(2, 8192)	(2, 8192)	(2, 8192)	(8, 8192)	(2, 8192)	(2, 8192)	(2, 8192)
Precinct size R3 (height, width)	(2, 8192)	(2, 8192)	(2, 8192)	(16, 8192)	(2, 8192)	(2, 8192)	(4, 8192)
Precinct size R4 (height, width)	(2, 8192)	(2, 8192)	(4, 8192)	(32, 8192)	(2, 8192)	(4, 8192)	(8, 8192)
Precinct size R5 (height, width)	(2, 8192)	(4, 8192)	(8, 8192)	(64, 8192)	(4, 8192)	(8, 8192)	(16, 8192)

minimal latency configuration. Using larger precincts allows the rate-distortion (R-D) optimization to result in a more optimal spatial variation of the effective quantization of the precinct, leading to better quality at the same bitrate at the expense of some additional latency. This alternative nonminimal latency configuration was selected to demonstrate HTJ2K's flexibility and also to achieve a middle ground between the other HTJ2K configurations and J2K ULL.

HTJ2K full-frame was also tested, using the same 9/7 wavelet with five levels as in JPEG 2000 full-frame, but instead using the HT block coder.

Like J2K, the HTJ2K standard is intended to be royalty-free, and the primary technology contributor to HTJ2K (Kakadu R&D Pty Ltd.) has made royalty-free declarations to the ITU and ISO.

VC-2

VC-2 (SMPTE ST 2042) is a DWT-based codec that uses a dead-zone quantizer and an interleaved form of exp-Golomb coding as a variable-length entropy code. DWT coefficients are reordered into "slices" to allow for low-latency transmission.

The tests described in this paper use the VC-2 reference code maintained on GitHub by the BBC. The configuration used was LeGall 5/3 wavelet, three-level depth, with slice width 16 and slice height 8. Note that the reference code requires a "16p2" format for input, so the `convert_to_16p2` tool in the VC-2 reference code GitHub repository was used to convert from YUV files.

At the time of publication of ST 2042, no notice had been received by SMPTE claiming patent rights essential to the implementation of the standard. As stated in Ref. 6, "VC-2 is believed to be free from patents and so may be used freely without royalty payments. This has been achieved in part simply as a result of the simplicity of the codec and partly through careful choice of nonpatented algorithms."

JPEG XS

JPEG XS (ISO/IEC 21122) is a low-latency, low-complexity wavelet codec optimized for visually lossless compression (as evaluated per ISO/IEC 29170-2) for both natural and synthetic images. JPEG XS uses a 5/3 wavelet filter and collects quantization indices of all bands contributing to a given spatial region of an image into a "precinct," an integer number of which is grouped into a "slice" that extends over the full width of the image. JPEG XS packets contain entropy codec information on a single precinct, line, and a subset of the bands within the precinct and line. A slice contains coefficients that can be entropy-decoded independently.

This test utilized the ISO/IEC 21122-5 JPEG XS Reference Software in the main and high profiles using "psnr" rate optimization mode. The high profile uses five horizontal and two vertical wavelet decomposition levels, one column, and uniform quantization. The main profile is similar to the high profile but only uses one vertical wavelet decomposition.

Patent statements were received on ISO/IEC 21122 from two companies, which stated they were willing to negotiate licenses under reasonable and nondiscriminatory terms and conditions.

IP Transport

JPEG 2000 codestreams can be transported in MPEG Transport Stream (TS) encapsulation. J2K full-frame and J2K ULL are clearly defined for MPEG TS over Realtime Transport Protocol (RTP) using VSF TR-01:2018. JPEG 2000 codestreams (including HTJ2K) can also be transported as video elementary flows over RTP using Internet Engineering Task Force Request for Comments (IETF RFC) 5371 "RTP Payload Format for JPEG 2000 Video Streams"; however, to date, existing implementations of RFC 5371 (such as Gstreamer[7]) have mainly been for J2K full-frame.

IETF RFC 8450 "RTP Payload Format for VC-2 High Quality (HQ) Profile" defines the transport of VC-2 over RTP. An IETF internet draft is also currently being developed for the RTP carriage of JPEG XS.[8]

SMPTE ST 2110-22 "Professional Media Over Managed IP Networks: Constant Bit-Rate Compressed Video" specifies parameters for RTP transport of constant bitrate compressed video in ST 2110 systems. It requires that an RTP payload format be registered in accordance with IETF RFC 4855 and that the packetization shall produce a constant number of bytes per frame and a constant number of RTP packets per frame.

Codec Latencies

We define latency as the delay between the input of a video sample into the compression system and the output of that video sample from the decompression system. We further define "algorithmic latency" as the theoretical minimal amount of time for compression and decompression operations. It generally depends on the area of the video image required to be used in transforms and other compression processing that can be independently coded from other areas of the image.

It should be recognized that algorithmic latency is not equal to and generally less than "realized latency" in actual systems, which includes the speed of computation of compression and decompression, as well as packetization and other data transport requirements between the compressor and decompressor.

Algorithmic latency largely depends on the codec specification, while realized latency can differ dramatically between implementations. Dedicated hardware implementations such as on FPGAs may deliver a realized latency quite close to the algorithmic one; however, software implementation on commercial off-the-shelf

TABLE 2. Latencies for codecs and configurations used in this paper.

Codec tested	Algorithmic Latency Lines	Algorithmic Latency ms (1080p/50p)	Algorithmic Latency ms (1080p/59.94)	Latency analysis source
HTJ2K 9/7, 0 VWT	5	0.093 ms	0.077 ms	kdu_v_compress analysis
JPEG XS Main	10	0.185 ms	0.154 ms	ISO/IEC 21122-2, Annex E
HTJ2K 5/3, 1 VWT	11	0.204 ms	0.170 ms	kdu_v_compress analysis
HTJ2K 9/7, 1 VWT	15	0.278 ms	0.232 ms	kdu_v_compress analysis
JPEG XS High	20	0.370 ms	0.309 ms	ISO/IEC 21122-2, Annex E
HTJ2K 5/3, 2 VWT	24	0.444 ms	0.371 ms	kdu_v_compress analysis
VC-2 HQ	28	0.519 ms	0.443 ms	Private communication with BBC
HTJ2K 9/7, 2 VWT	36	0.667 ms	0.556 ms	kdu_v_compress analysis
HTJ2K 5/3, 2 VWT	48	0.889 ms	0.741 ms	kdu_v_compress analysis
HTJ2K 9/7, 3 VWT	172	3.185 ms	2.657 ms	kdu_v_compress analysis
J2K ULL 9 Stripes	240	4.444 ms	3.707 ms	analysis lines required, plus equal time for CBR rate control
J2K ULL 4 Stripes	540	10.0 ms	8.342 ms	analysis lines required, plus equal time for CBR rate control
J2K Full-Frame	2160	40.0 ms	33.367 ms	analysis lines required, plus equal time for CBR rate control
HTJ2K Full-Frame	2160	40.0 ms	33.367 ms	analysis lines required, plus equal time for CBR rate control

(COTS) devices may differ dramatically from the dedicated hardware implementation, and software implementations may differ in realized latency based on hardware capabilities such as the number of cores or graphics processing units (GPUs). Realized latency in real-world devices is often as much as twice the algorithmic latency or more.

Note that in the description of the HTJ2K codecs, the abbreviation "VWT" is used for the number of "vertical wavelet transforms."

Table 2 shows latencies for codecs and configurations used in this paper, as well as the sources for the latency data. Note that for J2K ULL, JPEG 2000 full-frame, and HTJ2K full-frame, the assumption used in this paper is that algorithmic latency should reflect the required time to accumulate arriving video lines for analysis, as well as an equal amount of time to that for the compressed representation of those lines to be effectively rate controlled to a constant encoded bitrate through the leaky-bucket buffer model.

Test Material

Test sequences were all 1920 × 1080 resolution 4:2:2 sampled 10-bit Rec. 709 uncompressed material. The three sequences used were:

- College football (CFB)—A 60-sec clip at 59.94 frames/sec of a college football broadcast with scenes of the audience in the stands, wide view of the field, views of running action, a replay, graphics, and a close-up of a player.
- CrowdRun—A 10-sec clip at 50 frames/sec of a large number of runners in a park with trees and the sky in the background, from the "Sveriges Television (SVT) High Definition Multi Format Test Set."
- ParkJoy—A 10-sec clip at 50 frames/sec of a pan shot of people running through a park with many trees, from the "SVT High Definition Multi Format Test Set."

CFB is a continuous, multiscene segment of an actual broadcast production that aired on a U.S. television network in 2019 and was recorded uncompressed at the outside broadcast (OB) truck. CrowdRun and ParkJoy are test sequences that have been widely used in video and image compression testing over the last decade. The SVT clips[9] are based on a high-quality 65-mm film capture running at 50 frames/sec.

The SVT clips were distributed in their native 16-bit SGI RGB format, which is no longer widely used by the industry in 2020. The SGI files were converted into 16-bit TIF files using The Foundry's

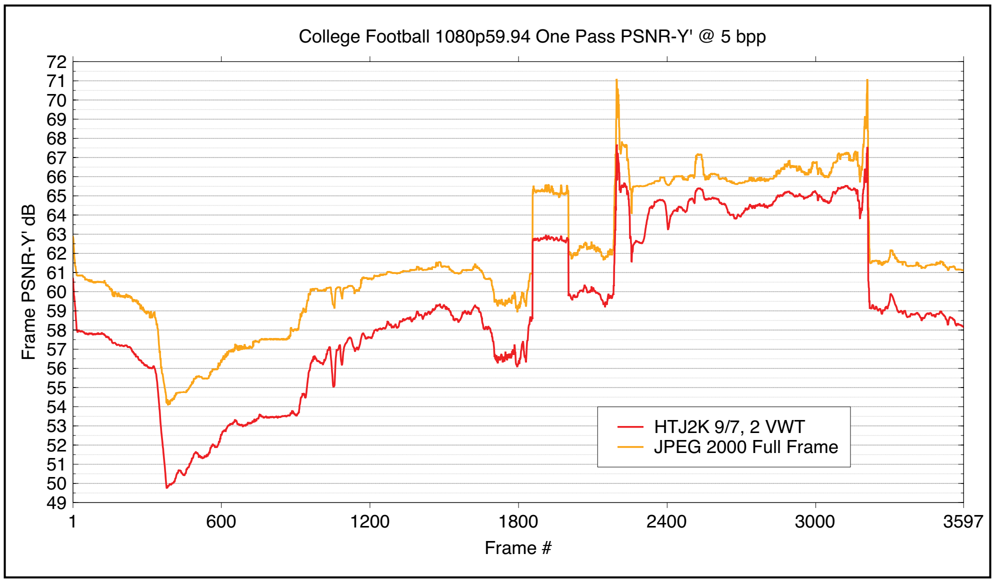

FIGURE 2. PSNR-Y' across CFB sequence frames with two codecs at 5 bpp.

Nuke software. The 16-bit TIF files were converted into a 10-bit 4:2:2 v210 MOV file using BlackMagic Design Davinci Resolve 16 software. The v210 MOV was converted into planar YUV 4:2:2 10-bit file format using FFMPEG software (i.e., `ffmpeg -i in.mov -c:v rawvideo -pix_fmt yuv-422p10le out_1920x1080p_50_10b_422.yuv`).

Test Procedure

Tests were performed on an Amazon Web Services r5.12xlarge EC2 instance initialized from the publicly available Ubuntu 18.04 LTS AMI. This instance type has 48 vCPUs and 384 GiB random-access memory (RAM). Media files were stored on a five-volume redundant array of independent disks (RAID) 0 gp2 drive, and a 350 GiB RAM drive was used for most processing and analyses.

Tests were performed at an integer number of bits per pixel (bpp) from 2 to 7. It should be noted that the VC-2 encoder was unable to produce a useful output at 2 bpp, so VC-2 was tested only between 3 and 7 bpp.

PSNR-Y' and SSIM-All metrics were calculated using FFMPEG's libavfilter ("-lavfi"). Bjøntegaard delta bitrate (BD rate) was calculated using The Department of Electronics and Informatics, Vrije Universiteit Brussel (ETRO's) Bjøntegaard metric implementation for Excel (https://github.com/tbr/bjontegaard_etro). Graphs were plotted using the Veusz scientific plotting package (https://veusz.github.io) that is Free Software.

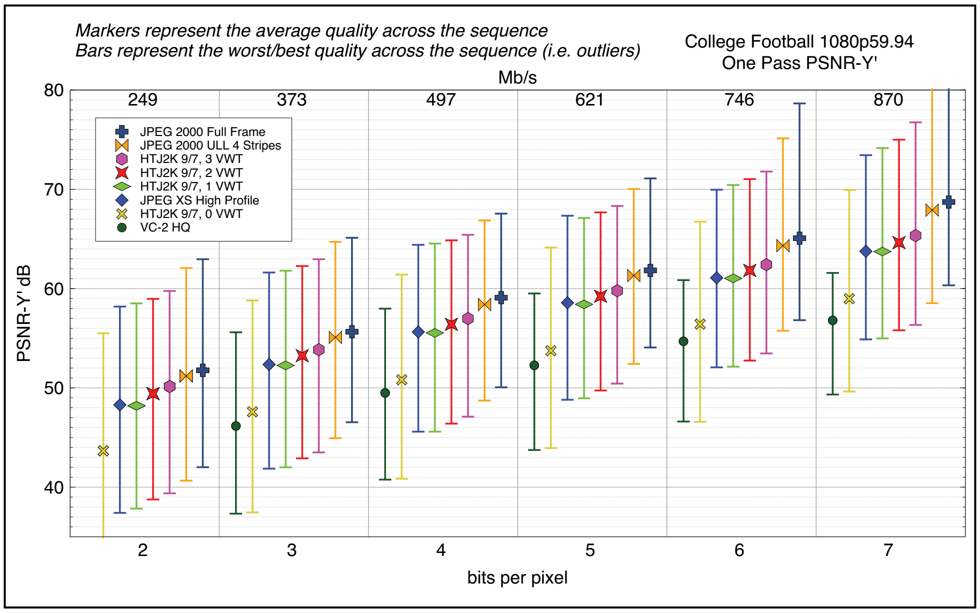

FIGURE 3. PSNR-Y' for CFB by data rate.

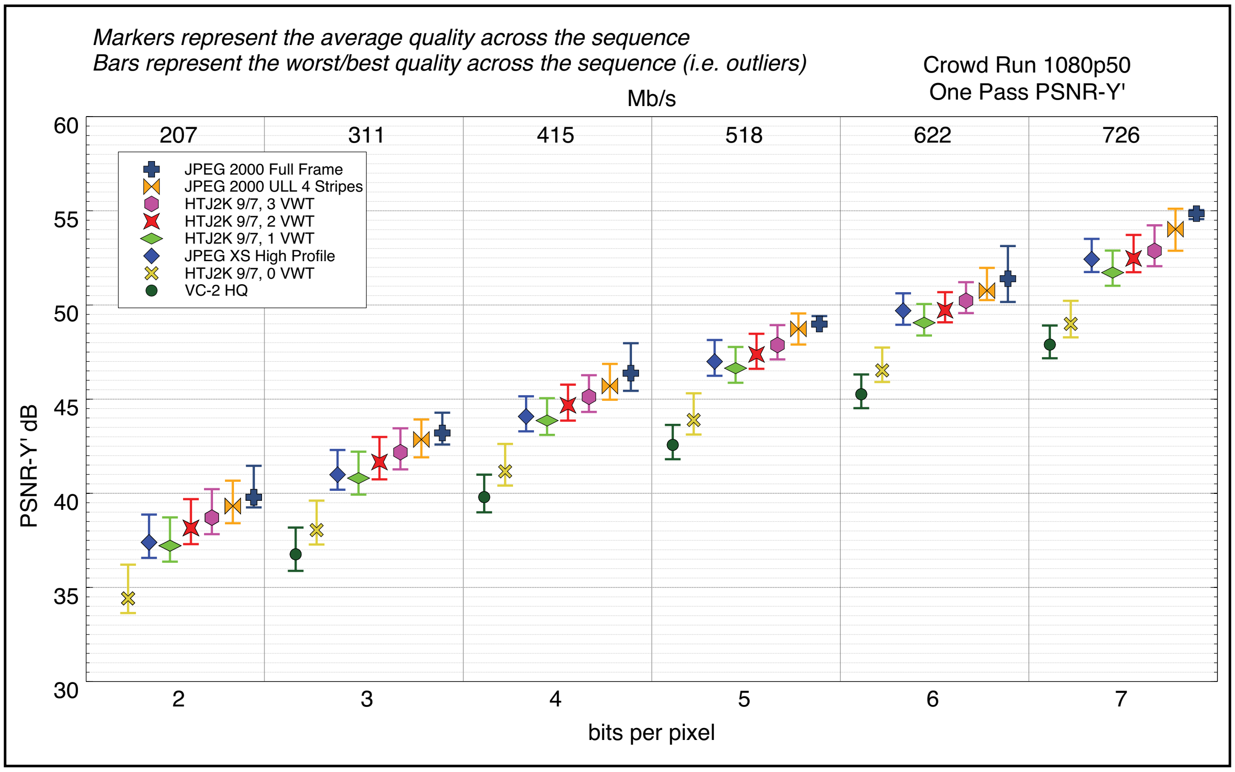

FIGURE 4. PSNR-Y' for CrowdRun by data rate.

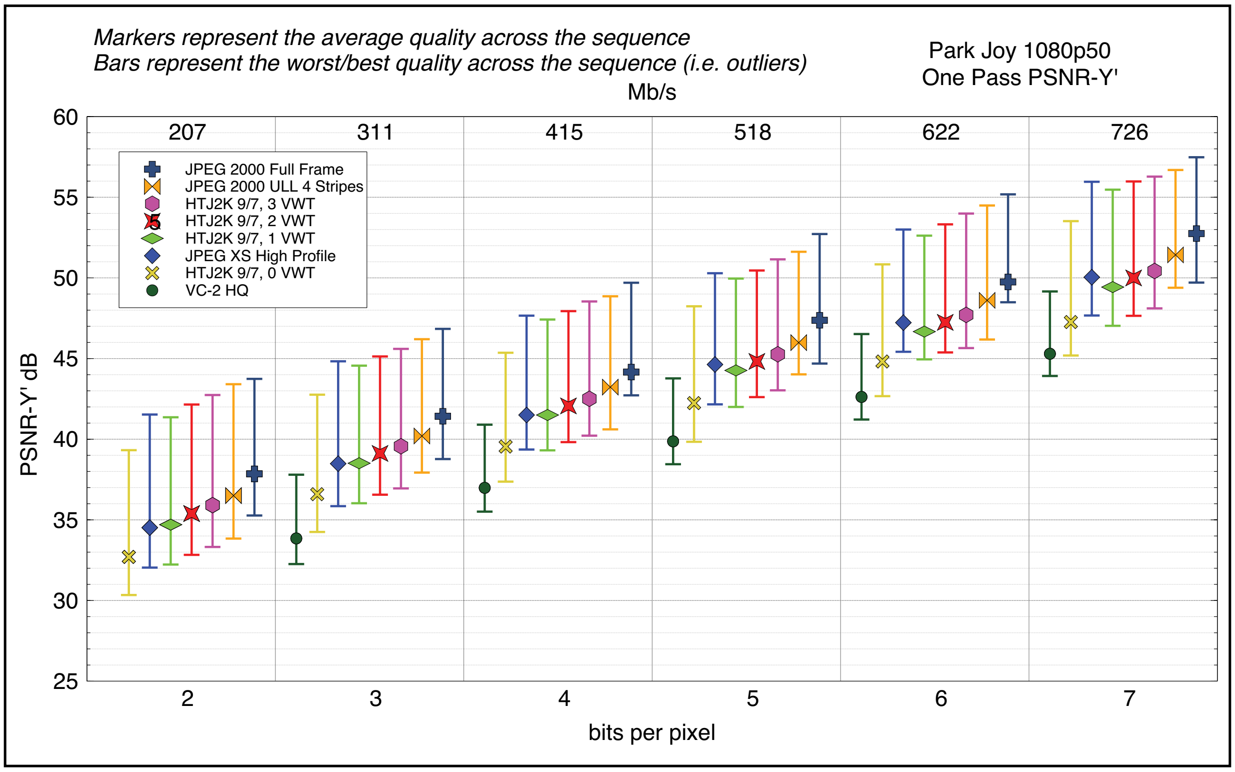

FIGURE 5. PSNR-Y' for ParkJoy by data rate.

Results

Figure 2 shows PSNR-Y' over the 3597 frames of the CFB sequence for HTJ2K 9/7, 2 VWT, and J2K full-frame. This shows how metrics like PSNR can change dramatically over time across a test sequence. Later figures will include the maximum value of a metric found across the many frames of the sequence, as well as the minimum value of the metric across the sequence.

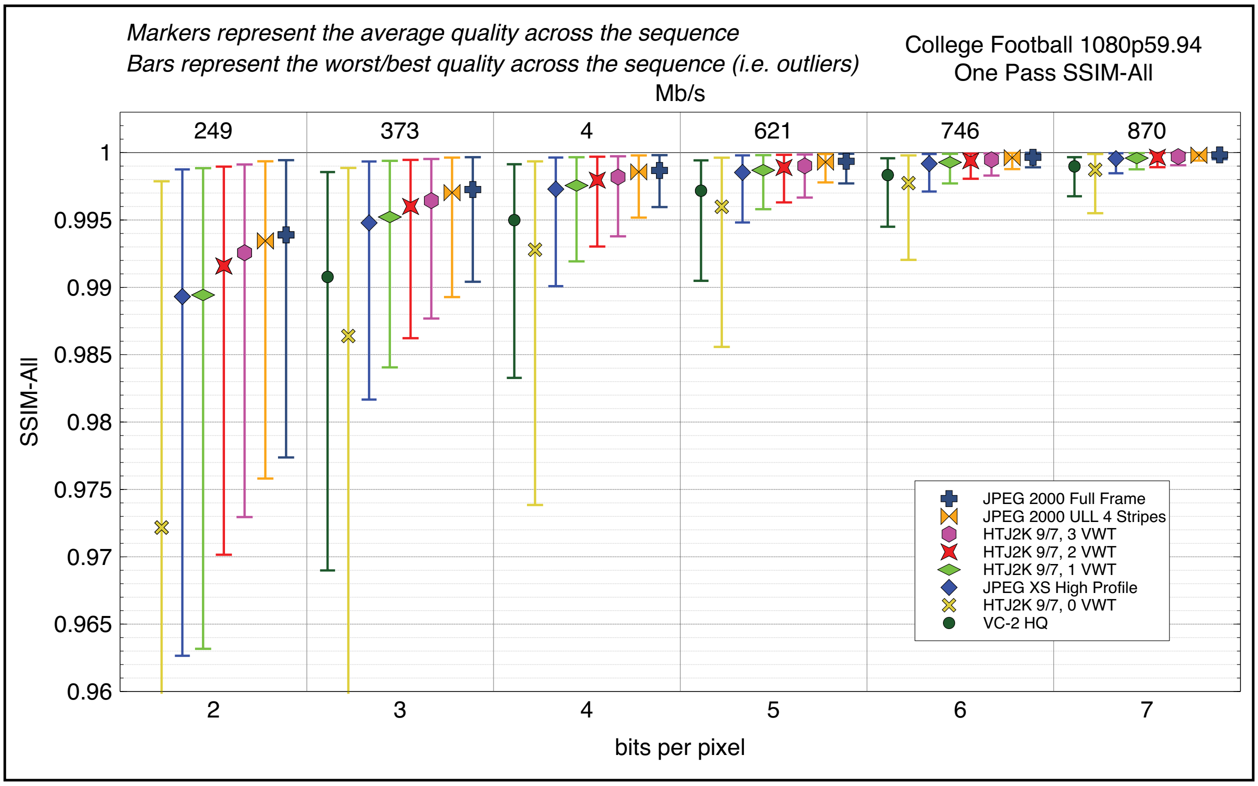

FIGURE 6. SSIM-All for CFB by data rate.

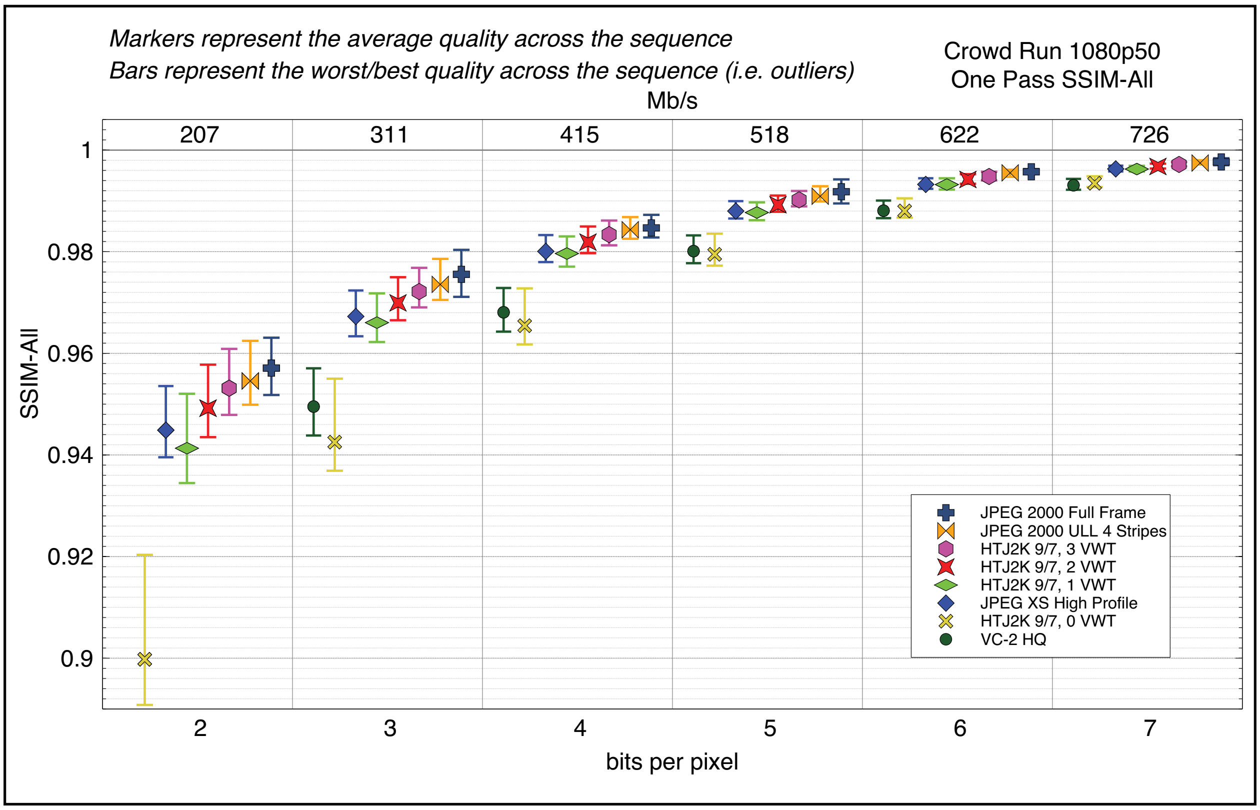

FIGURE 7. SSIM-All for CrowdRun by data rate.

Figures 3–5 show PSNR-Y' and **Figs. 6–8** show SSIM-All (structural similarity index across all components) for a single-pass compression of selected codecs. The average metric over all frames in a sequence is indicated by a marker, and bars below and above the marker show the minimum and maximum frame metric in the sequence.

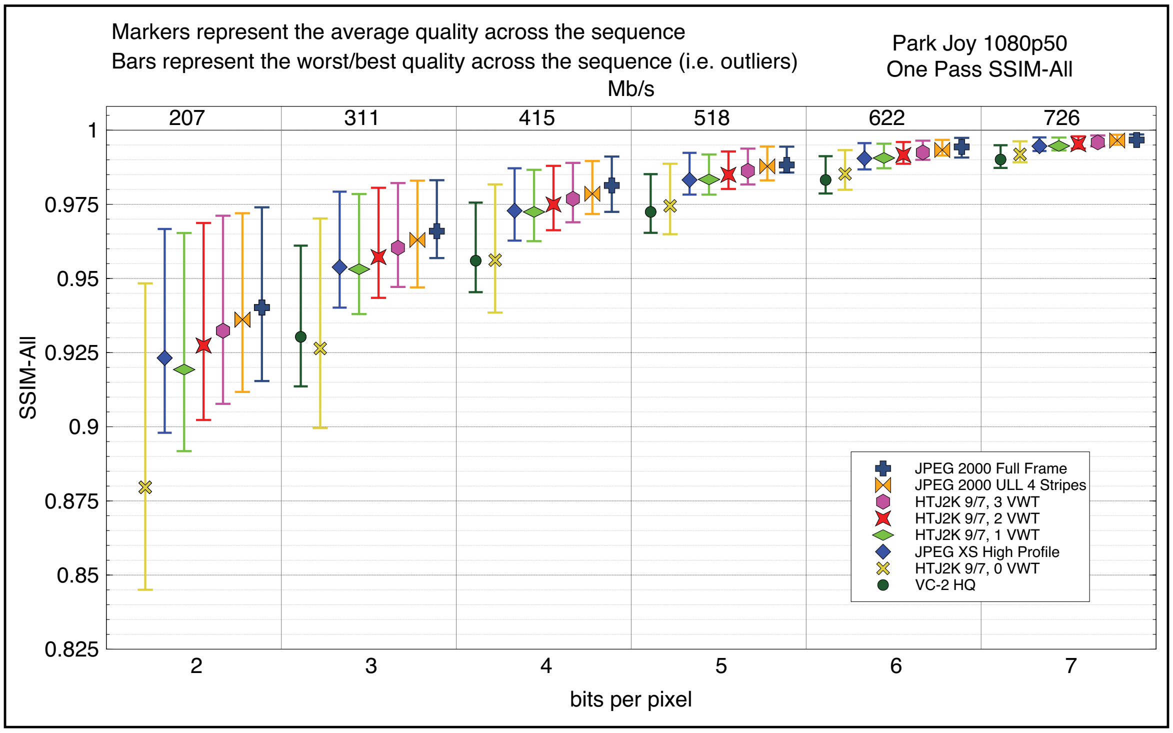

FIGURE 8. SSIM-All for ParkJoy by data rate.

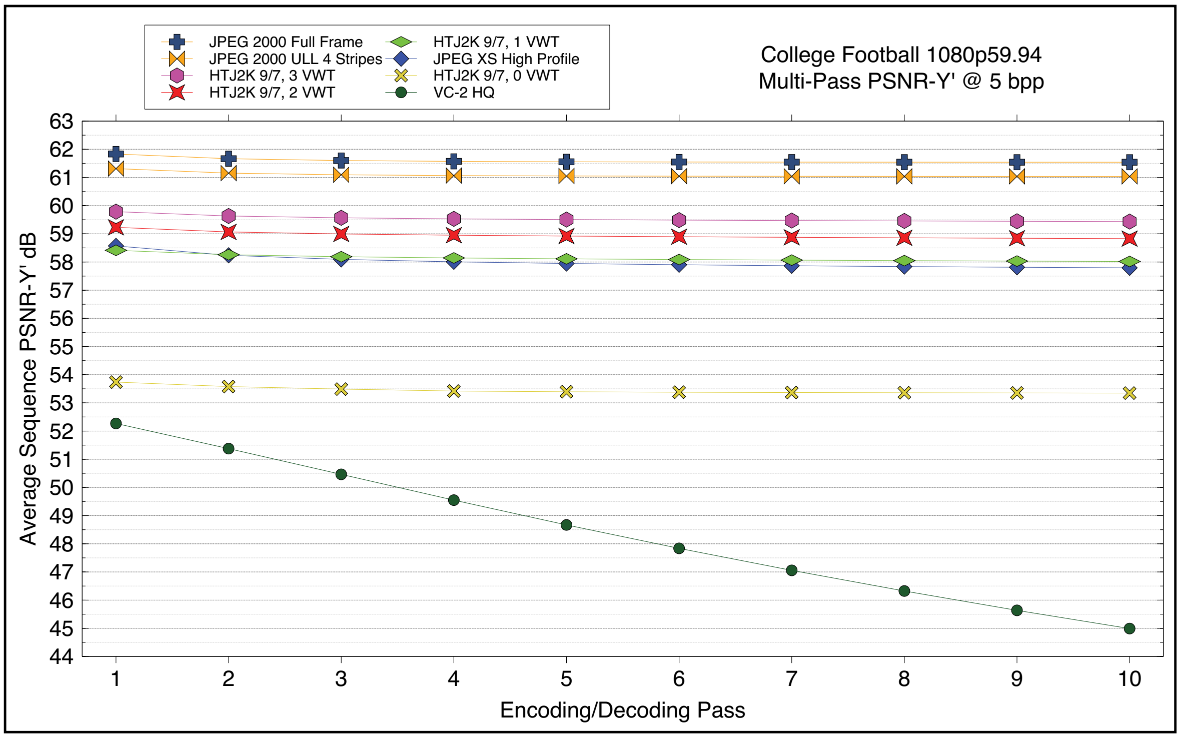

FIGURE 9. CFB multiple encoding/decoding pass PSNR-Y' at 5 bpp.

Figures 9–11 show the PSNR-Y' performance over multiple encoding/decoding passes of selected codec configurations at a data rate of 5 bpp.

Bjøntegaard model[10] rates (BD rates) for the three sequences based on data rates tested from 2 to 7 bpp are shown in **Figs. 12–14**. They are plotted against the algorithmic latency of the codec configurations. The PSNR-Y' BD rate calculates the average bitrate differences between two R-D curves obtained from the PSNR-Y' measurement when encoding at different bitrates. BD rates can be thought of as the amount of additional data rate required to achieve the equivalent quality metric. The anchor (i.e., 0% additional rate point) for these BD rates is J2K full-frame.

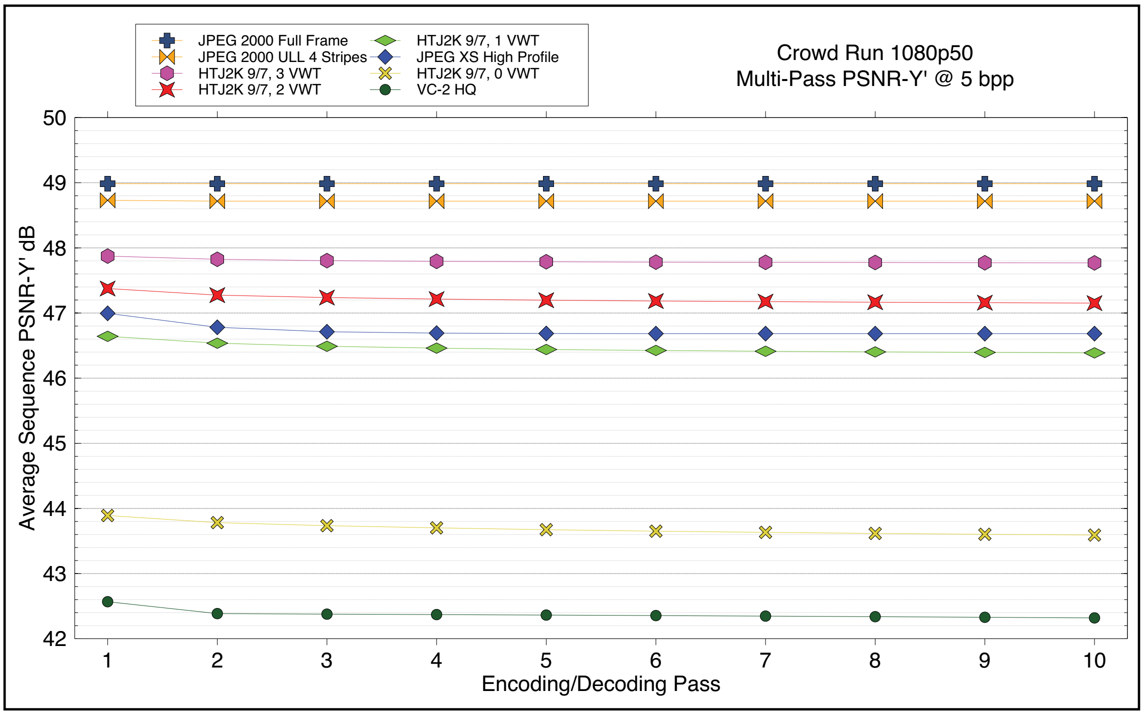

FIGURE 10. CrowdRun multiple encoding/decoding pass PSNR-Y' at 5 bpp.

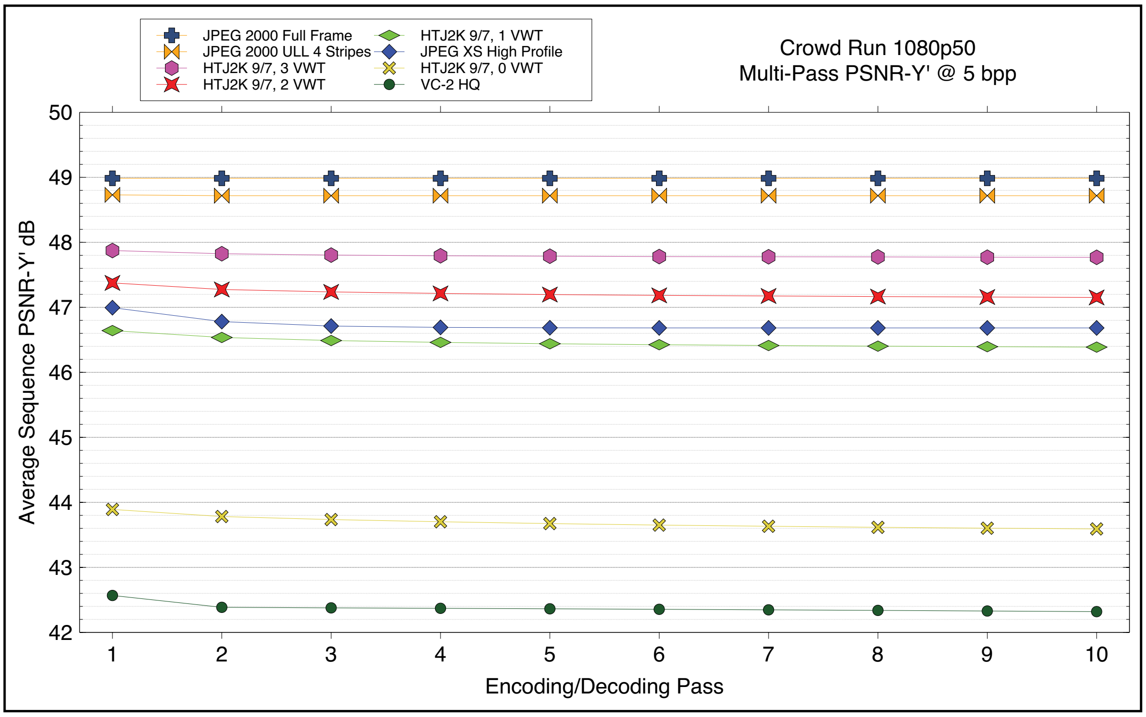

FIGURE 11. ParkJoy multiple encoding/decoding pass PSNR-Y' at 5 bpp.

Conclusion

These results show clearly the tradeoff between latency and quality at a particular data rate for these codecs. Most use cases for low-latency codecs involve trying to reduce the data rate for wide-area network (WAN) data lines or servers in a data center, especially attempting to move communication to the far more affordable regime of 1 Gb/s Ethernet as opposed to 10+ Gb/s Ethernet. In such cases, a reduction in data rates below 1 Gb/s achieves the desired goal. In situations where the data rate requires higher levels of compression, higher latency codec configurations may need to be used.

The codecs VC-2 HQ and HTJ2K 9/7, 0 VWT, seem to be the worst performing codecs tested. Interestingly, while HTJ2K 9/7, 0 VWT performed better than VC-2 HQ in PSNR-Y', it appeared to perform worse than

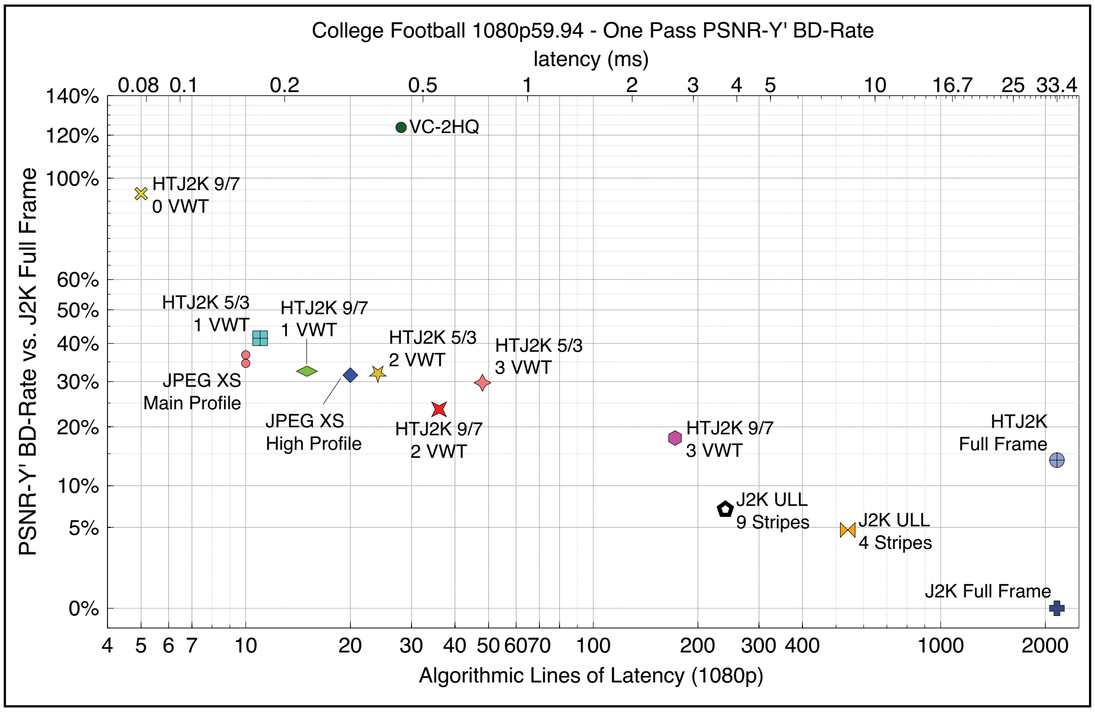

FIGURE 12. PSNR-Y' BD rate for CFB.

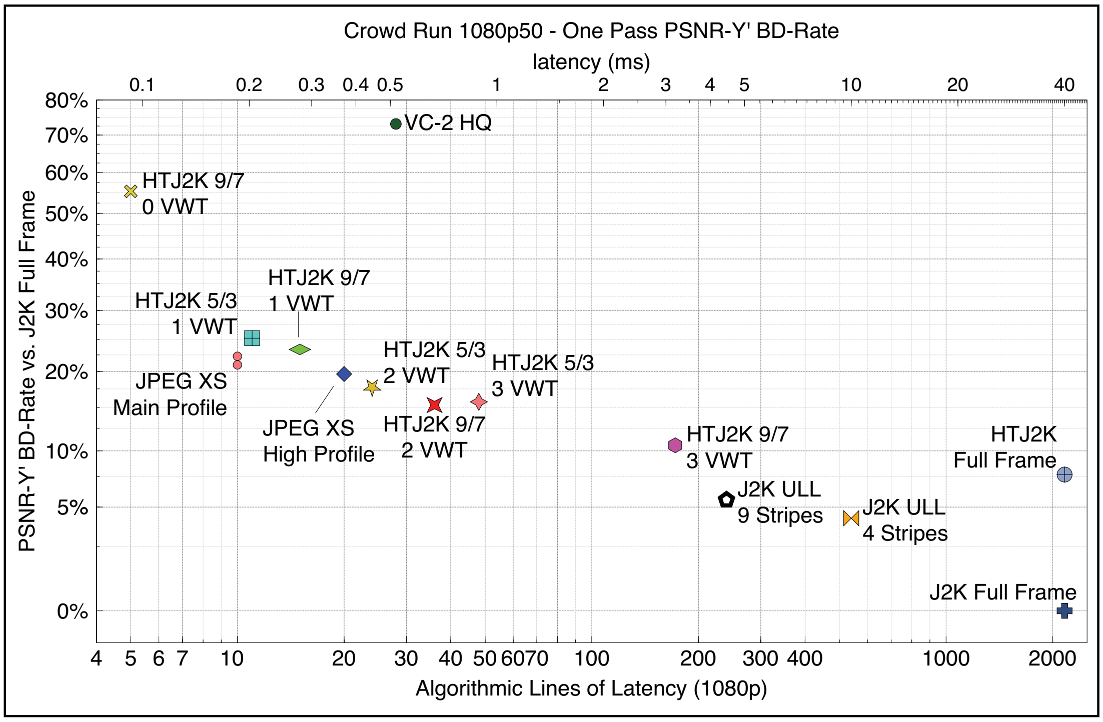

FIGURE 13. PSNR-Y' BD rate for CrowdRun.

VC-2 HQ in SSIM-All. VC-2 HQ performed much worse than other three-level transforms, and this may be due to its fairly simple entropy coding system. VC-2 HQ also had considerably more loss over multiple generations for the CFB sequence, which is long and has several segments of low visual complexity. It had less generational loss in the shorter and more consistently visually complex CrowdRun and ParkJoy sequences.

Except for the case of the VC-2 HQ codec with the CFB sequence, all codecs showed a fairly small generational quality loss, generally less than 1 dB PSNR-Y' over ten passes, and showing the most loss in the first four passes.

It is worth pointing out some interesting codec operating points: JPEG 2000 ULL delivers great quality for its latency; however, it is unclear if this codec can be efficiently implemented in software, and there are questions

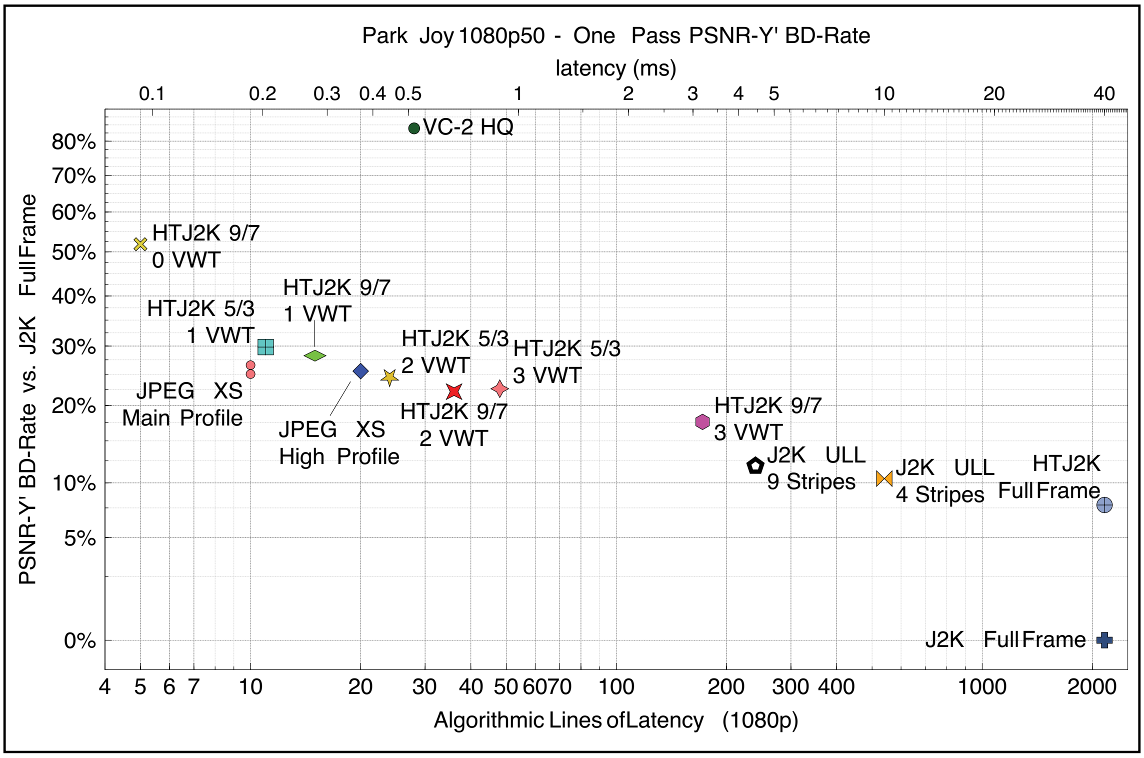

FIGURE 14. PSNR-Y' BD rate for ParkJoy.

about how to package it for use in ST 2110 systems. JPEG XS high profile and HTJ2K 5/3, 2 VWT should be considered for use cases requiring 20–30 lines of latency. JPEG XS main profile and HTJ2K 5/3, 1 VWT should be considered for ~10 lines of latency.

While "there is no such thing as a free lunch," these tests show that there is a wide spectrum of wavelet codecs and configurations to choose from, including the relatively new HTJ2K codec, and they all have different latency, quality, and computational characteristics. While there might not be a "free lunch," you can pick the "right lunch" for your particular application requirements.

Appendix: Command Line Encoding Parameters for Tests

VC-2

Code source: https://github.com/bbc/vc2-reference

```
"-x 1920 -y 1080 -f 4:2:2 -l 10 -k LeGall -d 3 -u 1 -a 2"
```

JPEG XS High Profile

Code source: ISO/IEC 21122-5 information technology—JPEG XS low-latency lightweight image coding system—Part 5: Reference software

```
"-p 4 -o psnr -w 1920 -h 1080 -d 10"
```

JPEG XS Main Profile

Code source: ISO/IEC 21122-5 information technology—JPEG XS low-latency lightweight image coding system—Part 5: Reference software

```
"-p 3 -o psnr -w 1920 -h 1080 -d 10"
```

JPEG 2000 Full-Frame

Code source:[3] kdu_v_compress & kdu_v_expand
For encoding and decoding, the "-precise" parameter was used. The "-frate" parameter for frame rate was also used for encoding.

```
"Sbroadcast={5,single,irrev} Qstep=0.0001 -no _ weights"
```

HTJ2K

Code source:[3] kdu_v_compress & kdu_v_expand
For all HTJ2K encoding and decoding, the "-precise" parameter was used. The "-frate" parameter for frame rate was also used for encoding.

HTJ2K 9/7, 0 VWT

```
"Corder=PCRL Clevels=5 Cprecincts={2,8192} Cblk={4,1024} Cdecomp=H(-) Qstep=0.0001 -no _ weights Scbr={1,3} Cmodes=HT Cplex={6,EST, 0.25,-1}"
```

HTJ2K 9/7, 1 VWT

```
"Corder=PCRL Clevels=5 Cprecincts={4,8192}, {2,8192} Cblk={4,1024} Cdecomp=B(-:-:-),H(-) Qstep=0.0001 -no _ weights Scbr={1,5} Cmodes=HT Cplex={6,EST,0.25,-1}"
```

HTJ2K 9/7, 2 VWT

```
"Corder=PCRL Clevels=5 Cprecincts={8,8192}, {4,8192},{2,8192} Cblk={4,1024} Cdecomp=B(-:-:-), B(-:-:-),H(-),H(-),H(-),H(-) Qstep=0.0001 -no _ weights Scbr={1,10} Cmodes=HT Cplex={6,EST, 0.25,-1}"
```

HTJ2K 9/7, 3 VWT

```
"Corder=PCRL Clevels=5 Cprecincts={64,8192},
{32,8192},{16,8192},{8,8192} Cblk={16,256} Cdecomp
=B(-:-:-),B(-:-:-),B(-:-:-),H(-),H(-),H(-)
Qstep=0.0001 -no _ weights Scbr={1,66} Cmodes=HT
Cplex={6,EST,0.25,-1}"
```

HTJ2K 5/3, 1 VWT

```
"Corder=PCRL Clevels=5 Cprecincts={4,8192},
{2,8192} Cblk={4,1024} Cdecomp=B(-:-:-),H(-)
Qstep=0.0001 Catk=2 Kkernels:I2=I5X3 -no _
weights Scbr={1,5} Cmodes=HT Cplex={6,EST,
0.25,-1}"
```

HTJ2K 5/3, 2 VWT

```
"Corder=PCRL Clevels=5 Cprecincts={8,8192},
{4,8192},{2,8192} Cblk={4,1024} Cdecomp=B(-:-:-),
B(-:-:-),H(-),H(-),H(-),H(-) Qstep=0.0001 Catk=2
Kkernels:I2=I5X3 -no _ weights Scbr={1,10}
Cmodes=HT Cplex={6,EST,0.25,-1}"
```

HTJ2K 5/3, 3 VWT

```
"Corder=PCRL Clevels=5 Cprecincts={16,8192},
{8,8192},{4,8192},{2,8192} Cblk={8,512} Cdecomp=B
(-:-:-),B(-:-:-),B(-:-:-),H(-),H(-),
H(-) Qstep=0.0001 Catk=2 Kkernels:I2=I5X3 -no _
weights Scbr={1,18} Cmodes=HT Cplex={6,EST,
0.25,-1}"
```

HTJ2K Full-Frame

```
"Sbroadcast={5,single,irrev} Qstep=0.0001
-no _ weights Cmodes=HT"
```

J2K ULL

Code source:[3] kdu_v_compress & kdu_v_expand
For J2K ULL encoding/decoding, the "-precise" parameter was used. The "-frate" parameter for frame rate was also used for encoding.

```
"Sbroadcast={5,single,irrev} Qstep=0.0001 -no _
weights Cblk={32,32} Clevels=4"
```

Acknowledgment

The authors wish to thank Dr. David Taubman for the use of the Kakadu software to perform some of the compression tests in this paper.

References

1. D. Taubman, A. Naman, and M. Reji, "FBCOT: A Fast Block Coding Option for JPEG 2000," *Proc. SPIE 10396, Appl. Digital Image Process. XL, 103960T*, pp. 1–17, 2017, doi: 10.1117/12.2273783.
2. T. Richter, J. Keinert, and S. Fößel, "Parallelization and Multithreaded Latency Constrained Parallel Coding of JPEG XS," *Proc. SPIE 11137, Appl. Digital Image Process. XLII, 111370J*, pp. 1–9, 2020, doi: 10.1117/12.2526917.
3. "Kakadu Software v8.0.5." Accessed: Mar. 2021. [Online]. Available: https://kakadusoftware.com
4. "Transport of JPEG 2000 Broadcast Profile Video in MPEG-2 TS Over IP," VSF TR-01:2018, Video Services Forum, 2018.
5. D. S. Taubman, A. Naman, M. D. Smith, and M. Regi, "High Throughput JPEG 2000 (HTJ2K): New Algorithms and Opportunities," *SMPTE Mot. Imag. J.*, 127(9):1–7, Sep. 2018, doi: 10.5594/JMI.2018.2859120.
6. T. Borer, "White Paper WHP 238: The VC-2 Low Delay Video Codec," BBC Research and Development, 2013.
7. "Gstreamer Open Source Multimedia Framework." Accessed: Mar. 2021. [Online]. Available: https://gstreamer.freedesktop.org
8. "RTP Payload Format for ISO/IEC 21122 (JPEG XS) draft-ietf-payload-rtp-jpegxs." Accessed: Mar. 2021. [Online]. Available: https://datatracker.ietf.org/doc/draft-ietf-payload-rtp-jpegxs/
9. L. Haglund, "The SVT High Definition Multi Format Test Set," Swedish Television, Stockholm, Sweden, 2006. Accessed: Mar. 2021. [Online]. Available: https://tech.ebu.ch/EBU_SVT_Public_Test_Sequences
10. G. Bjøntegaard, "Calculation of Average PSNR Differences Between RD-Curves, Technical Report VCEG-M33," ITU-T SG16/Q6, Austin, TX, USA, 2001.

About the Authors

Thomas Edwards is a principal solutions architect at Amazon Web Services (AWS), Seattle, WA, specializing in media and entertainment. Before joining AWS, he spent 20 years working for broadcasters PBS, Fox, and Disney on satellite content distribution, and digital TV projects, as well as exploring, proving out, evangelizing, and standardizing advanced media technology. He holds an MS degree in electrical engineering (EE) from the University of Maryland, College Park, MD, is a SMPTE Fellow, and was awarded the SMPTE Workflow Systems Medal for his work on live IP production technology.

Michael D. Smith has been working as a digital imaging and intellectual property consultant since 2003. He is currently a co-editor of the JPEG 2000 high-throughput image compression standard. In 2018, he received a screen credit for his work on *Mary Poppins Returns,* which was related to the color science used to integrate a traditional two-dimensional animation workflow with a modern Academy Color Encoding System (ACES) production pipeline. From 2013 to 2015, he was a co-chair of Blu-ray Disc Association's ultrahigh-definition (UHD)-Task Force (TF) Video Subgroup, which defined the video-related requirements for the UHD Blu-ray disc format. His work on more than 35 intellectual property matters related to infringement and validity of patents has resulted in payments of approximately $1.7 billion. He was an editor of the book "*3D Cinema and Television Technology: The First 100 Years*" published by SMPTE in 2011. He received BS and MS degrees in electrical engineering from the University of California at Los Angeles (UCLA), Los Angeles, CA, in 2001 and 2004, respectively.

Presented at the SMPTE 2020 Virtual Conference Event, 10–12 November 2020. Copyright © 2021 by SMPTE.