Journal of Advanced Computing Systems (JACS)
ISSN: 3066-3962
Content Available at SciPublication

Content-Adaptive Codec Selection Between AV1 and H.264: A Controlled Case Study of Bandwidth Savings Versus Latency at Matched Quality

Heyu Wang[1], *Phoebe Wu*[2]
[1]*Electrical and Computer Engineering, Rice University, TX, USA*
[2]*Artificial Intelligence, Northeastern University, MA, USA*
hw65@rice.edu

DOI: 10.69987/JACS.2024.40707

Keywords

AV1, H.264/AVC, content-adaptive streaming, codec selection, rate–distortion analysis

Abstract

AV1 often reduces bitrate relative to H.264/AVC, but the gain is operationally useful only when the extra computation does not violate live-encoding constraints. This paper presents a controlled case study of content-adaptive codec selection using an openly available 640×360, 120-frame Big Buck Bunny excerpt. The publicly distributed source asset available to us is a GIF excerpt, which is decoded once to a Y4M reference; the study therefore characterizes transcode behavior on an openly distributed sample rather than first-generation compression from a production master. FFmpeg/libx264 (preset=veryfast) and FFmpeg/libsvtav1 (preset=10) are evaluated across CRF sweeps using bitrate, PSNR, SSIM, and wall-clock encode/decode time. All derived quantities in this revision were recomputed from the tabulated measurements. Over the overlapping SSIM interval 0.935–0.961, cubic-fit Bjøntegaard delta-rate (BD-rate) is −26.5% for AV1 relative to H.264, whereas over the much narrower PSNR overlap 32.91–34.20 dB the PSNR-based BD-rate is +59.7%; the latter should be interpreted cautiously because the overlap is narrow and the AV1 PSNR curve is comparatively flat. At a near-matched-SSIM point (H.264 CRF 20 versus AV1 CRF 32), AV1 lowers bitrate from 2286.1 to 1806.0 kbps (−21.0%) while increasing encode time from 1.299 to 4.444 s (3.42×) and decode time from 0.600 to 0.826 s (1.38×). Segment-level analysis over four 1 s chunks shows that AV1 is bitrate-efficient throughout, but one higher-motion segment falls below the adopted SSIM floor. Under a 2.0 s per-segment encode-time budget and SSIM $\geq$ 0.94, a simple policy selects AV1 for three segments and H.264 for one, reducing bitrate from 2416.4 to 2295.9 kbps while increasing mean SSIM from 0.948 to 0.954 relative to an all-H.264 baseline. The main conclusion is practical: for this content and configuration, AV1 is attractive when processing budgets are relaxed, whereas H.264 remains the safer choice when real-time feasibility is the primary constraint.

Introduction

Video services increasingly operate multi-codec pipelines rather than committing to a single compression format across all content and delivery modes. H.264/AVC remains the default interoperability baseline because it is mature, broadly supported, and well documented [1], [2]. Newer codecs such as AV1 promise better compression efficiency, but they also introduce higher computational cost, especially at the encoder [4], [5]. The engineering problem is therefore not simply to ask which codec compresses better in the abstract, but which codec is preferable under a specific operational objective [23].

That objective depends strongly on the application. For video on demand, encoding is usually offline, so slower encoder settings can be tolerated if they reduce delivered bitrate. For live or interactive services, encode throughput contributes directly to end-to-end delay. In that regime, a bitrate reduction is helpful only if the encoder still produces segments fast enough for the streaming pipeline to remain near real time. This paper treats measured encode time as a proxy for live-latency

feasibility rather than as a complete glass-to-glass latency model [24].

Modern adaptive streaming systems already expose multiple representations per asset through MPEG-DASH or HTTP Live Streaming (HLS) manifests [12]–[15]. In principle, codec choice can be made per title, per device class, or per segment. This extends the logic of per-title optimization popularized in industry practice [18]: just as different content benefits from different ladders, different content can also benefit from different codecs [25].

Any matched-quality codec comparison also depends on the metric used to define "same quality." PSNR remains common because it is simple and historically entrenched, but SSIM often tracks perceived structural fidelity better [6], [7]. When two codecs rank differently under PSNR and SSIM, the resulting deployment decision can flip. A rigorous comparison must therefore report multiple metrics and be explicit about which metric drives any operational policy.

The present study is intentionally narrow and reproducible. We analyze a short public Big Buck Bunny excerpt [19] using fixed encoder configurations under FFmpeg [20]. Because the openly distributed excerpt available to us is a GIF sample, the study should be read as a controlled transcode case study on a public asset, not as a definitive source-master benchmark. That limitation is important, but the setup is still useful for illustrating how bandwidth savings and processing cost interact in a segment-based pipeline [26].

The contributions of the paper are fourfold. First, we present internally consistent full-clip rate–distortion and complexity results for FFmpeg/libx264 and FFmpeg/libsvtav1 at fixed operating points. Second, we recompute BD-rate from the tabulated measurements to summarize average bitrate differences under both PSNR and SSIM. Third, we analyze four 1 s segments using spatial information (SI) and temporal information (TI) descriptors to characterize content variation. Fourth, we translate the measurements into a simple budget-constrained selection rule that chooses the lower-bitrate feasible codec subject to an encode-time budget and an SSIM floor.

Method

Dataset and scope.

The source sequence is a short, openly available Big Buck Bunny excerpt [19] distributed as BigBuckBunny.gif. The clip resolution is 640×360 and the frame rate is 30 fps. The manuscript reports 120 frames, which implies a nominal frame-count duration of 4.00 s, while the file metadata reports 4.01 s. To keep all reported bitrate and throughput values internally consistent with the measured outputs, this revision retains 4.01 s for those calculations and treats the 0.01 s difference as a metadata-rounding discrepancy. The source GIF is decoded once into a Y4M (YUV 4:2:0, yuv420p) reference file so that both encoders operate on the same pixel-domain input. This step does not restore information lost in the GIF representation; it simply avoids repeated container-level decoding differences.

Segmentation.

To mimic segment-based streaming, the reference is partitioned into four nominal 1.0 s segments of 30 frames each: frames 0–29, 30–59, 60–89, and 90–119. Each segment is stored as an independent Y4M file and evaluated separately.

Encoders and operating points.

H.264/AVC is encoded through FFmpeg's libx264 wrapper [20], [21] with preset=veryfast. AV1 is encoded through FFmpeg's libsvtav1 wrapper [20], [22] with preset=10. For both codecs the study uses yuv420p input, GOP size 60, and no audio. Scene-cut detection is disabled for x264 to keep keyframe placement predictable. The H.264 RD sweep uses CRFs {18, 20, 21, 23, 28, 33}; the AV1 sweep uses CRFs {25, 30, 31, 32, 35, 40}. For segment-level matched-setting analysis, the paper compares H.264 CRF 20 with AV1 CRF 32 because those points lie close in average SSIM on the full clip.

Metrics and timing.

For each encoded output, average PSNR and average SSIM are computed against the Y4M reference using FFmpeg [20]. Bitrate is defined as output file size in bits divided by the reported duration of 4.01 s, and is reported in kbps. Encoding time and decoding time are measured as wall-clock elapsed time for a single run of each encode or decode job. Decode time is measured by decoding to a null sink so that rendering is excluded. Throughput is reported as speed relative to real time (×RT), computed as 4.01 s divided by the measured processing time.

BD-rate computation.

BD-rate is computed in the standard form by fitting a cubic polynomial to log-bitrate as a function of the quality metric and integrating over the overlapping quality range [6]. The overlap range is wide for SSIM (0.935–0.961) but narrow for PSNR (32.91–34.20 dB), so the PSNR-based BD-rate is informative but less stable and is interpreted cautiously.

Segment features and selection rule.

Each 1 s segment is characterized by spatial information (SI) and temporal information (TI). SI is measured as the standard deviation of Sobel-gradient magnitude on

the luma plane; TI is measured as the standard deviation of frame-to-frame luma differences. The operational selection rule is intentionally simple: for each segment, a candidate encode is feasible if its measured encode time is less than or equal to a budget B and its measured SSIM is at least Q_min. Among feasible candidates, the lower-bitrate codec is selected. If neither candidate is feasible, the policy falls back to H.264. Figure 6 additionally shows a TI-based visualization that explains the observed decisions, but that threshold plot is illustrative rather than a separately trained classifier.

Reproducibility notes.

This revision recomputes all derived statistics, tables, and figures from the internally consistent measurement values reported for the clip and its four segments. Because the timing measurements are single-run wall-clock observations on a short clip, they should be interpreted as descriptive case-study values rather than hardware-independent latency constants.

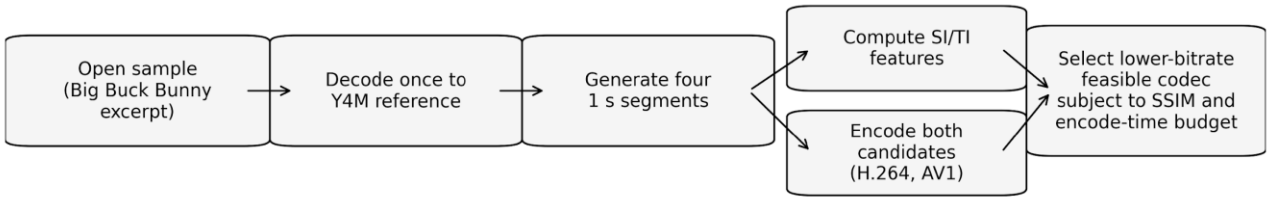

Figure 1. Content-adaptive codec selection pipeline used in the case study.

Table I. Dataset and preprocessing summary.

Item	**Value**
Source content	BigBuckBunny.gif (public sample file; excerpt of the open movie Big Buck Bunny)
Clip duration (reported)	4.01 s
Nominal frame-count duration	4.00 s (120 frames / 30 fps)
Resolution	640×360
Frame rate	30 fps
Frames	120
Original file size	2.91 MB
Original MD5	3c931dc0ce4bb84a4bb6666da811cb7d
Reference format	Y4M (YUV 4:2:0, yuv420p)
Reference file size	41.47 MB
Reference MD5	70b7871f980c3e0c10bbbb3568956d42
Segmentation	4 nominal segments × 1.0 s (30 frames each)

Table II. Encoder configurations and measurement protocol.

Component	Configuration
Software stack	FFmpeg 5.1.8; x264 core 164 r3095 baee400; SVT-AV1 Encoder Lib v1.4.1
H.264/AVC encoder	FFmpeg libx264; preset=veryfast; GOP=60; scenecut=0; pixel format=yuv420p; container=MP4
H.264/AVC CRFs	18, 20, 21, 23, 28, 33
AV1 encoder	FFmpeg libsvtav1; preset=10; GOP=60; pixel format=yuv420p; container=MKV
AV1 CRFs	25, 30, 31, 32, 35, 40
Metrics	Bitrate, PSNR, SSIM, encode wall time, decode wall time, processing speed, BD-rate
Timing protocol	Single wall-clock run per encode/decode point

Results and Discussion

Figures 2 and 3 plot the corresponding RD curves in PSNR and SSIM space. The H.264 sweep spans 207.1–3117.5 kbps, 28.78–38.19 dB PSNR, and 0.776–0.962 SSIM. The AV1 sweep spans 1085.2–2820.4 kbps, 32.91–34.20 dB PSNR, and 0.935–0.961 SSIM. The SSIM overlap is substantial, whereas the PSNR overlap is narrow.

That asymmetry matters. Under the sampled settings, H.264 attains higher PSNR at comparable bitrate over the limited PSNR-overlap window, while AV1 retains higher SSIM at comparable bitrate over most of the shared SSIM range. This is a reminder that matched-quality conclusions are metric-dependent rather than universal.

Table III. Full-clip RD and complexity results for H.264/AVC (x264, preset=veryfast).

CRF	Bitrate (kbps)	PSNR (dB)	SSIM	Enc time (s)	Dec time (s)	Enc speed (×RT)	Dec speed (×RT)
18.0	3117.5	38.19	0.962	1.38	0.694	2.91	5.78
20.0	2286.1	36.7	0.949	1.299	0.6	3.09	6.68
21.0	1946.9	35.98	0.942	1.282	0.673	3.13	5.96
23.0	1383.9	34.58	0.925	1.197	0.592	3.35	6.77
28.0	538.3	31.32	0.864	1.098	0.424	3.65	9.46
33.0	207.1	28.78	0.776	1.004	0.567	3.99	7.07

Table IV. Full-clip RD and complexity results for AV1 (SVT-AV1, preset=10).

CRF	Bitrate (kbps)	PSNR (dB)	SSIM	Enc time (s)	Dec time (s)	Enc speed (×RT)	Dec speed (×RT)
25.0	2820.4	34.2	0.961	4.909	1.013	0.82	3.96
30.0	2070.1	33.84	0.954	4.615	0.898	0.87	4.47

31.0	1937.7	33.76	0.953	4.31	1.007	0.93	3.98
32.0	1806.0	33.67	0.951	4.444	0.826	0.9	4.85
35.0	1486.5	33.4	0.946	4.026	0.974	1.0	4.12
40.0	1085.2	32.91	0.935	4.024	0.784	1.0	5.11

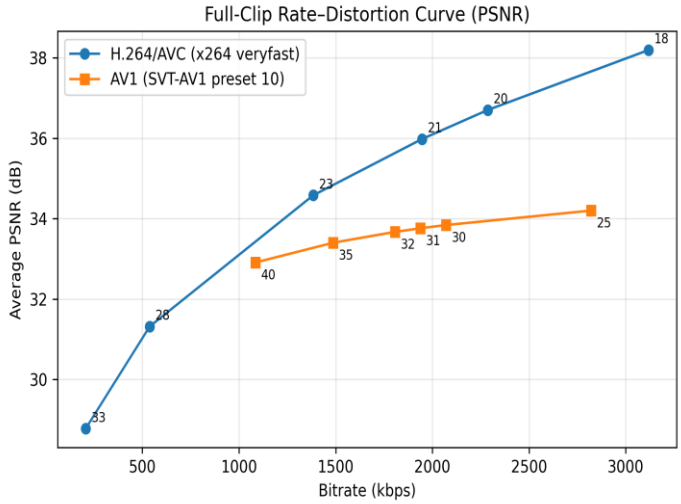

Figure 2. Full-clip rate–distortion curve in PSNR space.

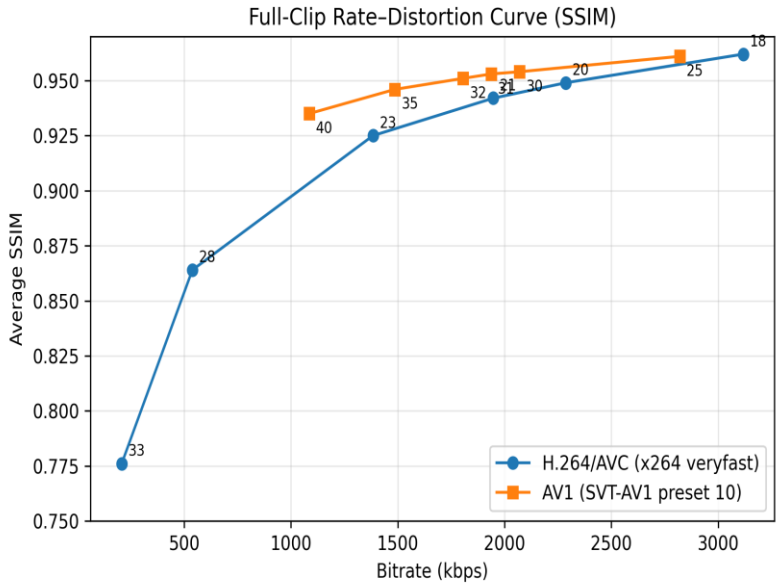

Figure 3. Full-clip rate–distortion curve in SSIM space.

Table V. BD-rate summary recomputed from the tabulated full-clip measurements.

Metric	BD-rate AV1 vs H.264 (%)	Integration range
PSNR	59.7	32.91–34.20 dB
SSIM	-26.5	0.935–0.961

Table V summarizes the average bitrate difference implied by the RD curves. Over the overlapping SSIM interval 0.935–0.961, AV1 achieves a recomputed BD-rate of −26.5% relative to H.264, which is consistent with the visual impression in Figure 3. Over the overlapping PSNR interval 32.91–34.20 dB, the recomputed PSNR-based BD-rate is +59.7%, indicating that AV1 requires more bitrate to match H.264's PSNR under these settings. Because that PSNR interval is only 1.29 dB wide and the AV1 PSNR curve is comparatively flat, the PSNR-based figure should be read as a narrow-window summary rather than as a broad statement about AV1 in general.

Table VI makes the trade-off concrete at a near-matched-SSIM operating point: H.264 CRF 20 and AV1 CRF 32. AV1 reduces bitrate by 21.0% and slightly improves SSIM (+0.002), but it lowers PSNR by 3.03 dB and raises encode time by 3.42×. In this case study, the bitrate advantage is therefore real but expensive in processing time.

Table VI. Near-matched-SSIM comparison (x264 veryfast CRF 20 versus SVT-AV1 preset 10 CRF 32).

Codec	Bitrate (kbps)	PSNR (dB)	SSIM	Enc time (s)	Dec time (s)	Enc speed (×RT)
H.264/AVC (CRF 20)	2286.1	36.70	0.949	1.299	0.600	3.08
AV1 (CRF 32)	1806.0	33.67	0.951	4.444	0.826	0.90
Δ (AV1 vs H.264)	−21.0%	−3.03 dB	+0.002	3.42×	1.38×	0.29×

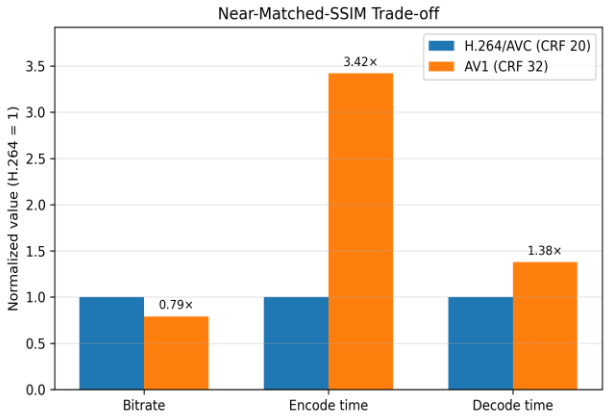

Figure 4. Near-matched-SSIM trade-off normalized to H.264/AVC.

The encode-time behavior is especially important for live deployment. Across the tested sweep, H.264 encodes at 2.90–3.99× real time, whereas AV1 runs at 0.82–1.00× real time. Decode time also favors H.264, but the difference is much smaller: 5.78–9.46× real time for H.264 versus 3.96–5.12× real time for AV1. On a desktop-class CPU, both are decodable faster than real time, but AV1 leaves less headroom.

The segment-level results show why a uniform codec choice can be suboptimal. Table VII reports SI/TI features for the four 1 s segments. The absolute SI/TI magnitudes depend on implementation details, but the ranking is informative: segment S3 has the highest temporal activity (TI mean 14.16, TI max 17.82), while S0–S2 are less dynamic. Table VIII shows that AV1 has lower bitrate than H.264 in every segment, yet its SSIM drops below H.264 only in S3. Thus, the more dynamic segment is precisely where the bitrate winner is not also the quality-floor winner.

Table VII. Segment-level content features computed from the reference sequence.

Segment	Time range (s)	SI mean	TI mean	SI max	TI max
S0	0.00–1.00	69.78	7.60	70.45	14.88
S1	1.00–2.00	69.49	7.28	70.31	11.86
S2	2.00–3.00	68.58	8.52	69.38	15.14
S3	3.00–4.00	65.47	14.16	67.58	17.82

Table VIII. Segment-level bitrate, quality, and processing cost at the matched settings.

Segment	Codec	Bitrate (kbps)	PSNR (dB)	SSIM	Enc time (s)	Dec time (s)
S0	H.264/AVC	2073.2	36.88	0.952	0.583	0.434
S0	AV1	2042.0	34.96	0.962	1.894	0.512
S1	H.264/AVC	2277.7	36.70	0.950	0.584	0.406
S1	AV1	2125.0	35.31	0.960	1.916	0.425
S2	H.264/AVC	2495.0	36.40	0.947	0.601	0.409
S2	AV1	2197.1	34.31	0.951	1.979	0.500
S3	H.264/AVC	2819.6	36.24	0.944	0.607	0.420
S3	AV1	2305.2	31.56	0.935	2.007	0.500

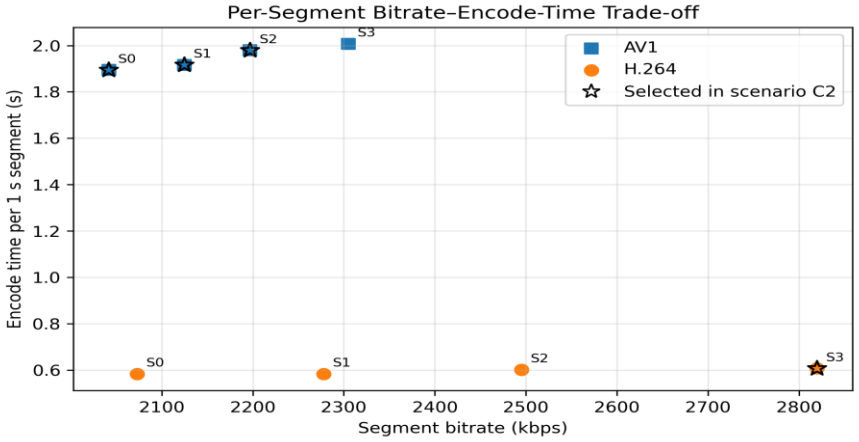

Figure 5. Per-segment bitrate versus encode time; starred points are selected in scenario C2.

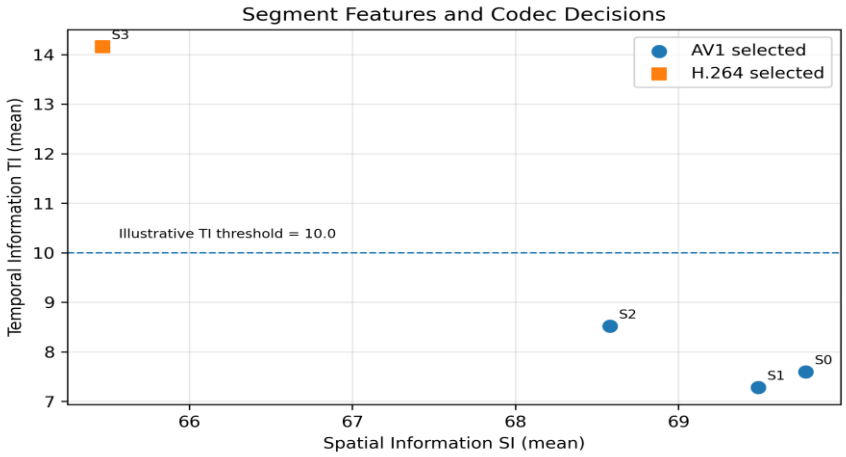

Figure 6. SI–TI feature map with the measured codec decisions for scenario C2.

Figure 5 plots the per-segment bitrate–encode-time trade-off. All AV1 points lie to the lower-bitrate but higher-computation side of the corresponding H.264 points. Figure 6 shows the same segments in SI–TI space. The dashed TI threshold is an explanatory visualization only, but it illustrates that the measured policy change coincides with the segment that has substantially higher temporal activity.

Table IX reports three budget scenarios. In C1, the encode-time budget is only 0.7 s per segment, so AV1 is infeasible throughout and the policy collapses to all-H.264. In C2, the budget increases to 2.0 s and the quality floor is SSIM $\geq$ 0.94; AV1 becomes feasible and preferable for S0–S2, but not for S3 because S3 simultaneously exceeds the 2.0 s budget (2.007 s) and misses the SSIM floor (0.935). In C3, the budget is relaxed to 10.0 s and the floor to 0.93, so all four segments switch to AV1.

Relative to the all-H.264 baseline, scenario C2 reduces average bitrate from 2416.4 to 2295.9 kbps (−5.0%) while improving mean SSIM from 0.948 to 0.954. Relative to all-AV1, scenario C2 accepts a 5.9% bitrate increase in exchange for a higher mean SSIM and lower total encode time. This is exactly the type of engineering compromise a mixed-codec production pipeline may wish to make.

A simple sequential live model further clarifies the operational effect. If each 1 s segment becomes available for encoding at the end of its capture interval and segments are processed sequentially, the all-AV1 pipeline accumulates about 4.80 s of processing delay by the end of segment S3, whereas the adaptive C2 policy reduces the final delay to about 3.40 s by switching the last segment to H.264. This is not a full streaming simulator, but it shows why encode throughput matters even when AV1 saves network bits.

Table IX. Codec-selection outcomes under encode-time and quality constraints.

Scenario	Encode-time budget per segment (s)	SSIM threshold	AV1 segments	H.264 segments	Overall bitrate (kbps)	Mean SSIM	Total enc time (s)
C1	0.7	0.94	0	4	2416.4	0.948	2.375
C2	2.0	0.94	3	1	2295.9	0.954	6.396
C3	10.0	0.93	4	0	2167.3	0.952	7.796

The overall lesson is not that one codec is categorically superior, but that the preferred codec depends on which constraint binds. For offline or near-live workflows, AV1's bitrate savings can be compelling. For strict real-time operation under the tested preset, H.264 remains easier to sustain because its encode speed is comfortably above real time. The segment-level analysis also shows that even a very small content window can contain a segment where the best bitrate codec is not the best operational choice. That observation motivates codec selection as a contextual decision rather than a fixed system-wide default.

Limitations

This is a deliberately small case study based on a single 4 s public excerpt, so it does not establish content-class averages or platform-wide rules. Different conclusions are plausible for sports, natural camera video, screen content, noisy sources, higher resolutions, and slower or faster encoder presets.

The public source asset used here is a GIF excerpt, not a mezzanine-quality production master. The Y4M conversion ensures a common encoder input, but it cannot recover color depth or detail already lost in the GIF. The results therefore characterize a controlled transcode benchmark on an open sample, which is useful for method comparison but weaker than a source-master study.

Timing values are based on single wall-clock runs for a short clip. They are suitable for illustrating order-of-magnitude differences between codecs, but they are too lightweight to support statistical confidence intervals or platform-independent claims.

Quality is evaluated with PSNR and SSIM only. Although SSIM is often more perceptually meaningful than PSNR [7], neither replaces subjective testing or stronger perceptual metrics. The divergence between PSNR-based and SSIM-based BD-rate in this study is itself evidence that metric choice materially affects the conclusion.

The selection rule uses measured per-segment encode time as a proxy for live-latency feasibility. It does not model packaging delay, network jitter, player buffering, hardware decode acceleration, or subjective QoE models such as P.1203 [10], [11]. A production-ready controller would need those additional terms.

Conclusion

This revised manuscript presents an internally consistent case study of content-adaptive codec selection between AV1 and H.264/AVC on a short Big Buck Bunny excerpt. Recomputed full-clip summaries show a clear metric dependence: AV1 achieves a −26.5% BD-rate over the overlapping SSIM range, yet a +59.7% BD-rate over the much narrower overlapping PSNR range. At a near-matched-SSIM point, AV1 cuts bitrate by 21.0% but increases encode time by 3.42×. Segment-level analysis then shows how a budget-constrained policy can recover most of the bitrate gain without accepting the weakest AV1 segment: under a 2.0 s per-segment encode-time budget and an SSIM floor of 0.94, the adaptive policy selects AV1 for three segments and H.264 for one, improving mean SSIM while still lowering bitrate relative to all-H.264. For this content, encoder configuration, and metric choice, the practical recommendation is straightforward: AV1 is attractive where processing slack exists, while H.264 remains the safer fallback when real-time feasibility dominates.

References

[1] T. Wiegand, G. J. Sullivan, G. Bjøntegaard, and A. Luthra, "Overview of the H.264/AVC video coding standard," IEEE Trans. Circuits Syst. Video Technol., vol. 13, no. 7, pp. 560–576, Jul. 2003.

[2] ITU-T, "Recommendation H.264: Advanced video coding for generic audiovisual services," International Telecommunication Union, Jun. 2019.

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

[4] Alliance for Open Media, "AV1 Bitstream & Decoding Process Specification," Version 1.0.0, 2018.

[5] J. Han et al., "A Technical Overview of AV1," arXiv:2001.11356, 2020.

[6] T. Bjøntegaard, "Calculation of average PSNR differences between RD-curves," ITU-T VCEG, Doc. VCEG-M33, Apr. 2001.

[7] Z. Wang, A. C. Bovik, H. R. Sheikh, and E. P. Simoncelli, "Image quality assessment: From error visibility to structural similarity," IEEE Trans. Image Process., vol. 13, no. 4, pp. 600–612, Apr. 2004.

[8] ITU-T, "Recommendation P.910: Subjective video quality assessment methods for multimedia applications," International Telecommunication Union, Apr. 2008.

[9] ITU-R, "Recommendation BT.500-13: Methodology for the subjective assessment of the quality of television pictures," International Telecommunication Union, Jan. 2012.

[10] ITU-T, "Recommendation P.1203: Parametric bitstream-based quality assessment of progressive download and adaptive audiovisual streaming services over reliable transport," International Telecommunication Union, Oct. 2017.

[11] W. Robitza, M. Garcia, D. Gómez, and A. Raake, "HTTP Adaptive Streaming QoE Estimation with ITU-T Recommendation P.1203," in Proc. 9th ACM Multimedia Systems Conf. (MMSys), 2018.

[12] I. Sodagar, "The MPEG-DASH Standard for Multimedia Streaming Over the Internet," IEEE Multimedia, vol. 18, no. 4, pp. 62–67, Apr. 2011.

[13] T. Stockhammer, "Dynamic adaptive streaming over HTTP—Standards and design principles," in Proc. 2nd ACM Multimedia Systems Conf. (MMSys), 2011, pp. 133–144.

[14] ISO/IEC, "ISO/IEC 23009-1: Dynamic adaptive streaming over HTTP (DASH)—Part 1: Media presentation description and segment formats," 2019.

[15] R. Pantos and W. May, "HTTP Live Streaming," RFC 8216, Aug. 2017.

[16] K. Spiteri, R. Urgaonkar, and R. K. Sitaraman, "BOLA: Near-optimal bitrate adaptation for online videos," IEEE/ACM Trans. Netw., vol. 28, no. 4, pp. 1698–1711, Aug. 2020.

[17] T.-Y. Huang, R. Johari, N. McKeown, M. Trunnell, and M. Watson, "A buffer-based approach to rate adaptation: evidence from a large video streaming service," in Proc. ACM SIGCOMM, 2014, pp. 187–198.

[18] Netflix Technology Blog, "Per-Title Encode Optimization," 2015. [Online]. Available: https://netflixtechblog.com/per-title-encode-optimization-7e99442b62a2

[19] Blender Foundation, "Big Buck Bunny," Open Movie Project, 2008. [Online]. Available: https://peach.blender.org/

[20] FFmpeg Developers, "FFmpeg Documentation." [Online]. Available: https://ffmpeg.org/documentation.html

[21] VideoLAN, "x264." [Online]. Available: https://images.videolan.org/developers/x264.html

[22] Alliance for Open Media, "SVT-AV1." [Online]. Available: https://gitlab.com/AOMediaCodec/SVT-AV1

[23] Jinyi Mu, Yifei Lu, and Michelle Smith, “LLM-Assisted Incrementality (Uplift) Modeling for Email Advertising: From Feature Interactions to Interpretable Audience–Creative–Channel Policies ”, JACS, vol. 3, no. 1, pp. 31–48, Jan. 2023, doi: 10.69987/JACS.2023.30103.

[24] Meng-Ju Kuo, Boning Zhang, and Haozhe Wang, “Tokenized Flow-Statistics Encrypted Traffic Analysis: Comparative Evaluation of 1D-CNN, BiLSTM, and Transformer on ISCX VPN-nonVPN 2016 (A1+A2, 60 s)”, JACS, vol. 3, no. 8, pp. 39–53, Aug. 2023, doi: 10.69987/JACS.2023.30804.

[25] Hanqi Zhang, “DriftGuard: Multi-Signal Drift Early Warning and Safe Re-Training/Rollback for CTR/CVR Models”, JACS, vol. 3, no. 7, pp. 24–40, Jul. 2023, doi: 10.69987/JACS.2023.30703.

[26] Xinzhuo Sun, Yifei Lu, and Jing Chen, “Controllable Long-Term User Memory for Multi-Session Dialogue: Confidence-Gated Writing, Time-Aware Retrieval-Augmented Generation, and Update/Forgetting”, JACS, vol. 3, no. 8, pp. 9–24, Aug. 2023, doi: 10.69987/JACS.2023.30802.