Benchmarking JPEG, MozJPEG, Jpegli, JPEG2000, JPEG XR, JPEG XL, WebP, HEIC, and AVIF-AOM/SVT: An Objective Quality and Coding Time Evaluation Across Encoder Configurations

Norishige Fukushima* Yoshihiro Maeda†, Yusuke Kameda‡, and Osamu Watanabe§
*Nagoya Institute of Technology, Nagoya, Japan
†Shibaura Institute of Technology, Tokyo, Japan
‡Sophia University, Tokyo, Japan
§Takushoku University, Tokyo, Japan

***Abstract*—In this paper, we propose a new benchmark dataset for still picture coding performance, named SPCP. SPCP is a large-scale benchmark dataset covering 10 codecs, 310 configurations, up to 101 quality parameters, 28 IQA metrics, and 3 computational times. The codecs are JPEG, MozJPEG, Jpegli, JPEG2000, JPEG XR, JPEG XL, WebP, HEIC, and AVIF-AOM/SVT. Also, we provide a public web interface for comparative analysis of still picture coding performance. Covering classical and newer codecs, SPCP provides a standardized testbed to support reproducible and fair comparison of image encoders. The full dataset is publicly available to accelerate codec development and evaluation in both academic and industrial communities. Our SPCP dataset is available at https://fukushimalab.github.io/spcp/.**

***Index Terms*—Still picture codec, IQA, dataset**

I. Introduction

High-performance image coding is an essential technology for information storage, and its standards are continually being updated. Among image compression formats, JPEG [1], [2] is a classic image compression format that still boasts the highest market share today. According to a report by W3Techs, 73.9% of websites use JPEG, while the newer encoding formats WebP and AVIF account for only 17.8% and 0.9%, respectively.

JPEG has lower encoding efficiency compared to newer formats, including WebP [3], JPEG XL [4], HEIC/HEIF [5], and AVIF [6], [7]. However, because of its high adoption rate, several JPEG tools have been proposed that maintain JPEG decoding compatibility while enabling high compression encoding. The libjpeg-turbo accelerates JPEG using SIMD from the original library developed by the Independent JPEG Group (IJG) (first released in 1991), and has been the official reference implementation of ISO/IEC/ITU-T since 2019. It also supports arithmetic coding extensions. MozJPEG [8] is a library forked from libjpeg-turbo by Mozilla in 2014. It is based on progressive JPEG (Annex G of ISO/IEC 10918-1) and can be optimized for multiple image quality assessment (IQA) metrics such as PSNR, PSNR-HVS [9], [10], SSIM [11], and MS-SSIM [12]. MozJPEG also supports arithmetic coding extensions. Jpegli [13] is a new library released by Google in 2024 as part of the JPEG XL subproject. Jpegli supports transcoding using XYB color space and JPEG XL entropy coding. Note that it does not support arithmetic coding and only supports Huffman optimization.

The next generation of JPEG standards includes JPEG 2000 [14] and JPEG XR [15], with JPEG XL [4] being the latest addition. Additionally, there are encoding formats developed based on intra-coding from video encoders, such as VP8-based WebP [3], H.265/HEVC-based HEIC [5], AV1-based AVIF (AOM) [6], and AVIF (SVT-AV1) [7], which applies scalable video technology. These formats achieve higher coding efficiency without the limitations of JPEG bitstreams. Furthermore, WebP and AVIF are supported by all standard browsers, while JPEG XL and HEIC are only supported by Safari; however, Firefox is expected to support JPEG XL.

The important factors for image coding are image quality, compression ratio, and encoding/decoding speed. Thus, numerous encoders offer multiple encoding options and quality tuning options for several IQA metrics, allowing them to handle the trade-off. However, there are not enough papers that comprehensively evaluate the various combinations of images, encoders, coding options, and IQA metrics. Most studies are limited to demonstrating a few specific combinations.

In this paper, we propose a large-scale benchmark dataset, named *Still Picture Coding Performance (SPCP)*, which comprehensively evaluates various compression options and evaluation metrics for multiple encoders listed in Table I. SPCP maintains the results of encoding 31,000 times per image for 310 configurations, with up to 101 different quality parameters (QPs). Additionally, 28 IQA metrics are evaluated, including 10 grayscale IQA metrics, 10 simplified color extensions of grayscale IQA, 7 color metrics, and 1 video metric. The IQA metrics used are as follows: Grayscale IQA (PSNR-Y, IW-PSNR [16], PSNR-HVS [9], [10], SSIM [11], MS-SSIM [12], IW-SSIM [16], VIF [17], GMSD [18], FSIM [19], NLPD [20]), Color IQA (PSNR-RGB, MDSI [21], Butteraugli (p=1,2,3,

This work was supported by JSPS KAKENHI (JP24K02960, 25K22830).

max) [22], [23], SSIMULACRA2 [24]) and VMAF [25], a video IQA, were used as intra-coding. Note that when using grayscale IQA for color image evaluation, each channel is converted to YUV in the BT.709 color space, and IQA is performed on each channel, resulting in an 8:1:1 ratio. Also, encoding, decoding, and total times are evaluated. Including bit-per-pixel (bpp) information, SPCP provides $1,001,920 = 101 \times 310 \times (28 + 3 + 1)$ data points in 32-bit floating-point format, i.e., 3.91 MB per image. In addition, we provide an interface for both web and local data.

II. SPCP BENCHMARK DATASET

The following sections describe the three aspects of this benchmark, specifically the image codec, IQA, and dataset.

A. Image Codec

We utilize JPEG-compatible encoders such as libjpeg-turbo, MozJPEG, and Jpegli, as well as advanced image encoders including JPEG 2000, JPEG XR, JPEG XL, WebP, HEIF, and AVIF (AOM, SVT). Most codecs support color spaces such as YUV420, 444, 440, and 422. Although aspect ratio differences in downsampling have a minor impact on PSNR, they have a significant effect on subjective quality. Therefore, YUV440 and 422 were not adopted for SPCP. The following sections provide an overview of each codec.

1) JPEG: Since the JPEG standard was introduced in 1992, several new JPEG standards for image coding have been developed. JPEG LS (1998) [26] aimed at lossless compression. JPEG 2000 (2000) [14] targeted high compression, JPEG XR (2009) [15] is based on Windows Media Photo / HD Photo, considering both high compression and lightweight design. JPEG XT (2015) is a functional JPEG extension to handle higher bits, HDR, lossless coding, and alpha channels, but not for coding efficiency. JPEG XS (2019) is for visually lossless coding with low latency and complexity; thus, it is not a direct competitor to efficient image codecs. JPEG XL (2021) [4], [27] and its Jpegli subproject improved coding efficiency. Among these standards, only JPEG, JPEG XT, and Jpegli are compatible with each other. First, we will discuss JPEG-compatible codecs of libjpeg-turbo, MozJPEG, and Jpegli.

libjpeg-turbo is one of the fastest JPEG libraries optimized by SIMD. It can select baseline (B) and progressive (P) encoding. Its entropy coders are standard Huffman (S), optimized Huffman (O), and arithmetic coding (A). For progressive coding, optimized tables are used for the progressive mode. Additionally, we can use arithmetic coding implemented on MozJPEG using transcoding (MA), which has a different performance. Also, we can use new entropy coding, such as ANS, with JPEG XL transcoding (XL) without any degradation. Color spaces include YUV420, YUV422, YUV440, and YUV444, but YUV420 and YUV444 are adopted. The valid coding combinations for SPCP are 9: BS, BO, BA, BMA, BXL, PO, PA, PMA, and PXL. Note that if the only difference in the encoding settings is the entropy coder, the output image will not change; thus, IQA calculation can be omitted in JPEG-compatible coding.

MozJPEG is based on progressive coding and cannot use standard Huffman, requiring at least optimized Huffman. Furthermore, the quantization table can be tuned for PSNR, PSNRHVS, SSIM, and MS-SSIM. Note that arithmetic coding and optimized Huffman coding have different DCT coefficients due to the tuning process. For focusing on the entropy coder, we disable BA and PA of the native JPEG encoder options, but we use the transcoder of MA, which is also the MozJPEG product. There are 6 valid combinations for SPCP: BO, BMA, BXL, PO, PMA, and PXL. The color space is the same as in libjpeg-turbo. Note that MozJPEG is essentially based on progressive coding, and the baseline coding has not been sufficiently tuned. MozJPEG produces different output even when the options for the cjpeg and jpegtran command-line tools are the same. That is, the arithmetic coding results with cjpeg and the Huffman coding results with cjpeg followed by the arithmetic coding with jpegtran are different. However, the libjpeg-turbo case is the same. This is because the DCT table optimization differs between the *-arithmetic* and *-optimize* options in cjpeg, resulting in different output images. However, the difference is minimal. Since the output images differ, it is necessary to re-execute IQA. In this case, arithmetic encoding was performed using the transcoder.

Jpegli supports standard and optimized Huffman coding, but does not support arithmetic coding. However, JPEG XL, the original project of Jpegli, provides excellent transcoding for JPEG. Additionally, the arithmetic coding can be supported by MozJPEG's transcoding. Note that the baseline coding only supports standard Huffman coding; thus, we need transcoding of optimized Huffman for the baseline mode. Therefore, the valid combinations are 7: BS, BO, BMA, BXL, PH, PMA, and PXL. In addition to YUV, the XYB color space can be used. Note that 420 downsampling cannot be used in XYB; thus, it is limited to 444. Furthermore, Jpegli is optimized for Butteraugli [22], [23], similar to JPEG XL.

The DCT in JPEG [28], [29] includes the integer DCT LLM [30], the fast AAN [31], and the float DCT. The float version has slightly better coding performance than the integer one. However, circuit implementation and other factors make integer-based DCT more likely, and integer-based DCT is faster. Therefore, in this experiment, integer-based conversion was used in libjpeg-turbo and MozJPEG. Note that Jpegli only supports JDCT_FLOAT; thus, float was used.

2) Next JPEG: **JPEG 2000** quantizes and arithmetic-codes frequency coefficients obtained by wavelet transform with EBCOT. The Kakadu implementation was used for SPCP. The input color space used was YUV420 and 444. Additionally, two encoding configurations were handled regarding the quantization of luminance and chrominance: one where they are treated equally and another where luminance is prioritized. In total, 4 coding options are handled.

JPEG XR employs a lifting scheme of the photo core transform for frequency transform and also utilizes the photo overlap transform for deblocking filtering. JPEG XR uses YCoCg-R color space, not YUV, and we use 420 and 444 downsampling. For overlap parameters, we used 0, 1, and 2.

TABLE I: Image codecs and implementations. Conditions per image: the number of colorspaces × metric-tunes × modes. For JPEG, we add a default setting (De), and red colored modes need transcoders. In total, 310 conditions per image.

Codec	Library	Year	Color	Tunes	QP	Encoder modes	Conditions per image
JPEG [1], [2]	libjpeg-turbo 3.1.2	1992	420, 444	PSNR	0–100	BS, BO, BA, BMA, BXL, PO, PA, PMA, PXL, De	$19 = 2 \times 1 \times 9 + 1$
MozJPEG [8]	MozJPEG v4.1.5	2014	420, 444	PSNR, SSIM, PSNRHVS, MS-SSIM	0–100	BO, BMA, BXL, PO, PMA, PXL, De	$49 = 2 \times 4 \times 6 + 1$
Jpegli [13]	Jpegli v0.11.1	2024	420, 444, xyb	Butteraugli	1–100	BS, BO, BMA, BXL, PO, PMA, PXL, De	$22 = 3 \times 1 \times 7 + 1$
JPEG 2000 [14]	kakadu v86R	2000	420, 444	PSNR	1–99	no weight, yuv weight	$4 = 2 \times 1 \times 2$
JPEG XR [15]	jxrlib 2019.10.9	2009	420, 444	PSNR, SSIM	0.0–1.0	overlap (0,1,2)	$12 = 2 \times 2 \times 3$
JPEG XL [4]	libjxl 0.11.1	2021	xyb	Butteraugli	0– 99	VarDCT (2 – 10), Modular (2 – 10), *1: debug mode	$18 = 1 \times 1 \times (2 \times 9)$
WebP [3]	libwebp 1.6.0	2010	420, 420s	PSNR	0–100	0 – 6	$14 = 2 \times 1 \times 7$
HEIC [5]	libx265 4.1, libde265 v1.0.16	2013	420, 444	PSNR, SSIM	0–100 (51)	0, 1, 2, 5, 6, 7, 9	$28 = 2 \times 2 \times 7$
AVIF (AOM) [6]	libaom 3.13.1, dav1d 1.5.1	2019	420, 444	PSNR, SSIM, IQ, SSIMULACRA2, Butteraugli	0–100 (64)	0 – 9	$100 = 2 \times 5 \times 10$
AVIF (SVT) [7]	SVT-AV1 3.1.2, dav1d 1.5.1	2020	420	PSNR, SSIM, IQ, VQ	0–100 (64)	0 – 10	$44 = 1 \times 4 \times 11$

The quantized tables are tuned for PSNR and SSIM In total, we have 12 coding options.

JPEG XL is the latest image coding based on the JPEG standardization. Various encoding options can be specified, grouped under the *–effort* switch, which specifies a trade-off between speed and accuracy: 1 (fast) to 10 (slow). Note that 1 is reserved for debugging, hence the inclusion of 9 options. Additionally, JPEG XL has two entirely different encoding modes: VarDCT and Modular. The color space is XYB, and while there is currently no option for downsampling, xyb444 can be specified. As a result, a total of $2 \times 9 = 18$ options are available. Note that the modular mode produces the same transformed coefficient; thus, IQA scores are unchanged under different effort switches, which accelerates IQA computing.

3) Video Intra-based Encoder: Here, we explain WebP, HEIF, and AVIF, which are video intra-based encoders.

WebP is based on VP8 intra coding and offers various options, but the *–m* switch allows you to specify a trade-off between speed and accuracy. The range is 0 (fast) to 6 (slow). The color space is fixed at YUV420, and 444 cannot be used. Instead, YUV420sharp (420s in short) has been proposed, which increases the frequency. YUV420sharp is a YUV conversion and can, in principle, be used with any other library. However, since it was introduced specifically for WebP, it is evaluated here solely in the context of WebP. In total, 14 encoding options are supported.

HEIC (High-Efficiency Image Codec) is based on H.265/HEVC intra coding and offers various options, with a preset to specify the trade-off between speed and accuracy. The range is 0 (fast) to 9 (slow), but the presets 2–4 and 7-8 have the same meaning for still image coding, respectively. When optimizing the coding coefficients, we can tune them for PSNR or SSIM as an IQA metric. The used color spaces are YUV420 and 444. In total, 28 encoding options are supported. HEIC is a default codec in HEIF (High Efficiency Image File Format) containers, but HEIF can include various codecs. The intra codec for HEVC images is often referred to as HEIF, but for clarity, we will refer to it as HEIC here. For the HEIF container, we use libheif 1.19.8.

AVIF (AV1 Image File Format) is based on the AV1 (AOMedia Video 1) intra coding and offers various options, allowing users to specify a trade-off between speed and accuracy via switches. The range is 0 (slow) to 9 (fast), but the order is reversed. The libavif accepts the parameter of speed (10), but it is the same for speed (9) for the AOM codec. Note that the AVIF's coding time is notably longer than the other codecs. AVIF has an accelerated implementation called SVT, which can be distinguished and described as AVIF (AOM) and AVIF (SVT). Note that AVIF (SVT)'s option has a range of 0-10, one more than AOM. Furthermore, currently, only downsampled versions of the YUV420 color space are supported for SVT. AOM has tune options for PSNR, SSIM, IQ (image quality), SSIMULACRA2, and Butteraugli. SVT's options are PSNR, SSIM, IQ, and VQ (video quality). Therefore, encoding combinations are 100 for AOM and 44 for SVT. For the AVIF container, we use libavif1.3.0.

B. IQA metrics

We use three types of IQA metrics: grayscale, color, and video. For grayscale IQA, we use PSNR, IW-PSNR [16], PSNR-HVS [9], [10], SSIM [11], MS-SSIM [12], IW-SSIM [16], GMSD [18], FSIM [19], and NLPD [20]. Most of these metrics are used for JPEG XL and JPEG AI [32]. These metrics are used for grayscale signals; thus, we use the IQA metrics by converting color images to Y channel images. Handling color information, we convert the colorspace using BT.709, compute IQA for each channel, and then blend them with 8:1:1. For color IQA, we use PSNR-RGB, MDSI [21], Butteraugli (p = 1, 2, 3, max) [22], [23], SSIMULACRA2 [24]. For PSNR-RGB, compute MSE at once as a 3-channel signal is a 1D vector. MDSI is the native color metric. Butteraugli and SSIMULACRA2 are developed for coding distortion and can natively handle color images. For Butteraugli, we can change the pooling norm, and 4 types are used. For video IQA, we use VMAF [25] as an intra codec, which is a grayscale metric. Additionally, we include encoding/decoding/total time as metrics.

SPCP requires IQA computation many times; thus, we optimize IQA as fast as possible, such as through SIMD optimization [33] and convolution optimization [34], [35].

C. Image dataset

In SPCP, image datasets were collected in 4 contexts: common image, IQA, super-resolution (SR), and coding datasets. Table II shows the list, which can support a variety of images. Classics are typical image datasets, such as Barbara, Baboon, and Pepper. Kodak is also used in IQA datasets LIVE [36], TID2013 [37], [38], MIDD-CAP [39], [40], but the sizes differ because they have been cropped for subjective assessment. Additionally, the coding datasets for the Challenge on Learned Image Compression (CLIC) datasets of test (not validation), covering 2021, 2023, 2024, and 2025, have been encoded.

TABLE II: Dataset. * average size of variable-sized images.

Name	Context	Images	Size
Classics	Common	20	$547 \times 547^*$
Kodak	Common	24	768×512
LIVE [36]	IQA	29	$597 \times 597^*$
TID2013 [38]	IQA	25	512×384
CSIQ [41]	IQA	30	512×512
CID:IQ [42]	IQA	23	800×800
KADID-10K [43]	IQA	81	512×384
URBAN100 [44]	SR	100	$879 \times 879^*$
MANGA109 [45]	SR	109	$982 \times 982^*$
MIDD-CAP [40]	Coding	24	496×496
CID22 [24]	Coding	250	512×512
MCL-JCI [46]	Coding	50	1920×1080
CLIC2021	Coding	60	$1620 \times 1620^*$
CLIC2022	Coding	30	$1711 \times 1711^*$
CLIC2024	Coding	32	$1694 \times 1694^*$
CLIC2025	Coding	30	$1695 \times 1695^*$
Total	-	917	$961 \times 961^*$

III. Experimental Results

Since the results are enormous, please refer to the URL for more detailed results: https://fukushimalab.github.io/spcp/. Figure 1 shows the web interface that provides convenient access to SPCP. On the website, users can select datasets (Tab. II) through a dropdown menu and choose individual images, for which thumbnail previews are displayed on mouse hover. Encoder configurations can be selected by checkboxes, and evaluation metrics can be chosen from a dropdown list. The graph interface supports interactive zooming and panning, logarithmic scaling or axis inversion, and delta plots against a reference codec. It also provides Bjøntegaard delta rate (BD-rate) [47] and BD-quality (e.g., BD-PSNR), with flexible specification of both the metric range and the plotting range. In addition, users may enable or disable averaging of results across images at the same QP value. After setting, we can see RD plots (Fig. 2) or BD-plots from selected reference (Fig. 3).

Figure 4 shows the scatter plots of the average time of each QP and BD-rate of SSIMURACLA2 from the JPEG 420 baseline for each condition (one of the CLIC2025 images). The points circled in AVIF (AOM/SVT) represent the SSIMULACRA2 tune (blue) and IQ tune (green), each optimized for SSIMULACRA2. AVIF (AOM) implementation achieves the highest coding efficiency. Considering computation time, JPEG2000 also offers a good balance. Compared to JPEG 420 baseline coding, encoding time differed by up to 3100 times (5.6ms / 17,500ms), with a BD-rate (SSIMULACRA2) improvement of -51.6%. Decoding speed differences were limited to a maximum of about 25 times (6.5ms/162ms), with HEIC being the slowest (plot omitted). AVIF was about 4.5 times and relatively fast. The total time required to encode and decode all conditions for a single image (2048×1216) using an AMD Threadripper 3970X (16-core/32-thread) was 20.1 hours. Calculating the entire dataset would take approximately one year; thus, multiple computers were used for computation.

IV. Conclusion

This paper proposed SPCP, a benchmark dataset for the comprehensive evaluation of image codecs. SPCP comprised a large-scale collection of encoding results covering 310 configuration settings, 101 QPs, and 10 codecs across multiple image datasets. As future work, we will support emerging codecs such as JPEG AI, H.266/VVC intra, WebP2, and AVIF-AV2, and incorporate new IQA metrics.

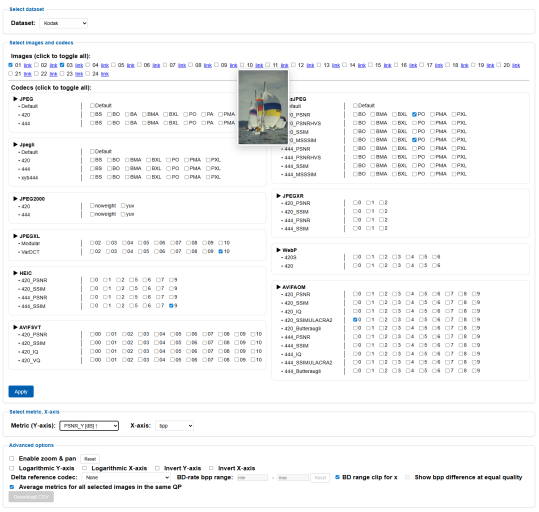

Fig. 1: Web Interface of SPCP.

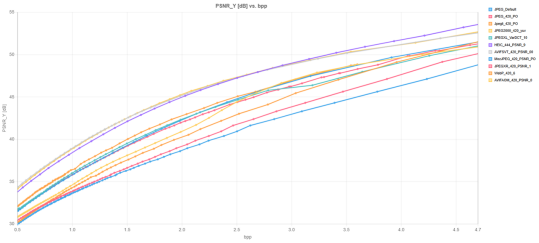

Fig. 2: An example of RD plots.

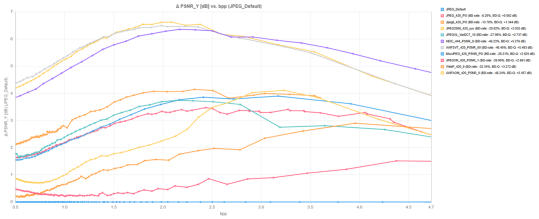

Fig. 3: An example of BD-plots.

References

[1] G. K. Wallace, "The jpeg still picture compression standard," *Communications of the ACM*, vol. 34, no. 4, pp. 30–44, 1991.

[2] D. G. Morgan, "libjpeg-turbo: Turbocharged jpeg image codec," in *Applications of Digital Image Processing XXXIX*. SPIE, 2016.

[3] Google WebM Project, "libwebp," https://github.com/webmproject/libwebp, 2010, accessed: 2025-05.

[4] J. Alakuijala *et al.*, "Jpeg xl image coding system," in *Proc. Picture Coding Symposium (PCS)*, 2021.

[5] G. J. Sullivan, J.-R. Ohm, W.-J. Han, and T. Wiegand, "Overview of the high efficiency video coding (hevc) standard," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 22, no. 12, pp. 1649–1668, 2012.

[6] D. Mukherjee *et al.*, "The latest open-source royalty-free video codec," *IEEE Transactions on Broadcasting*, vol. 64, no. 3, pp. 552–567, 2018.

[7] F. Kossentini, H. Guermazi, N. Mahdi, C. Nouira, A. Naghdinezhad, H. Tmar, O. Khlif, P. Worth, and F. B. Amara, "The svt-av1 encoder: overview, features and speed-quality tradeoffs," in *Applications of Digital Image Processing XLIII*, vol. 11510, 2020, pp. 469–490.

[8] Mozilla Corporation, "Mozjpeg project," https://github.com/mozilla/mozjpeg, 2014, accessed: 2025-05.

[9] K. O. Egiazarian, J. Astola, N. N. Ponomarenko, V. V. Lukin, F. Battisti, and M. Carli, "A new full-reference quality metrics based on hvs," in *Proc. International Workshop on Video Processing and Quality Metrics for Consumer Electronics*, 2006.

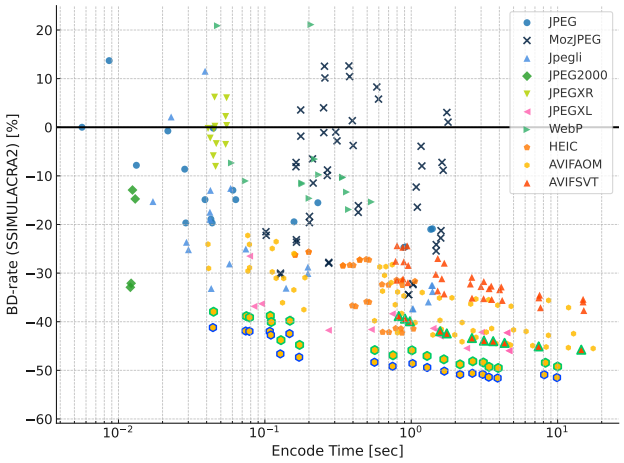

Fig. 4: Scatter plot of encoding time and BD-rate (SSIMULACRA2) for each condition.

[10] N. Ponomarenko, F. Silvestri, K.Egiazarian, M. Carli, and V. Lukin, "On between-coefficient contrast masking of dct basis functions," in *Proc. International Workshop on Video Processing and Quality Metrics for Consumer Electronics*, 2007.

[11] Z. Wang, A. C. Bovik, H. R. Sheikh, and E. P. Simoncelli, "Image quality assessment: from error visibility to structural similarity," *IEEE Transactions on Image Processing*, vol. 13, no. 4, pp. 600–612, 2004.

[12] Z. Wang, E. P. Simoncelli, and A. C. Bovik, "Multiscale structural similarity for image quality assessment," in *Proc. Asilomar Conference on Signals, Systems Computers*, vol. 2, 2003, pp. 1398–1402 Vol.2.

[13] J. Alakuijala *et al.*, "Jpegli: Enhancing jpeg with new encoding techniques," *arXiv preprint arXiv:2401.11974*, 2024.

[14] D. Taubman and M. Marcellin, *JPEG2000: Image Compression Fundamentals, Standards and Practice*. Springer, 2002.

[15] G. J. Sullivan *et al.*, "The jpeg xr image coding standard," *IEEE Signal Processing Magazine*, vol. 29, no. 1, pp. 146–170, 2012.

[16] Z. Wang and Q. Li, "Information content weighting for perceptual image quality assessment," *IEEE Transactions on Image Processing*, vol. 20, no. 5, pp. 1185–1198, 2011.

[17] H. R. Sheikh and A. C. Bovik, "Image information and visual quality," *IEEE Transactions on Image Processing*, vol. 15, no. 2, pp. 430–444, 2006.

[18] W. Xue, L. Zhang, X. Mou, and A. C. Bovik, "Gradient magnitude similarity deviation: A highly efficient perceptual image quality index," *IEEE Transactions on Image Processing*, vol. 23, no. 2, pp. 684–695, 2014.

[19] L. Zhang, L. Zhang, X. Mou, and D. Zhang, "Fsim: A feature similarity index for image quality assessment," *IEEE Transactions on Image Processing*, vol. 20, no. 8, pp. 2378–2386, 2011.

[20] V. Laparra, J. Ball'e, A. Berardino, and E. P. Simoncelli, "Perceptual image quality assessment using a normalized laplacian pyramid," *Electronic Imaging*, vol. 28, pp. 1–6, 2016.

[21] H. Z. Nafchi, A. Shahkolaei, R. Hedjam, and M. Cheriet, "Mean deviation similarity index: Efficient and reliable full-reference image quality evaluator," *Ieee Access*, vol. 4, pp. 5579–5590, 2016.

[22] J. Alakuijala, R. Obryk, O. Stoliarchuk, Z. Szabadka, L. Vandevenne, and J. Wassenberg, "Guetzli: Perceptually guided jpeg encoder," *arXiv preprint arXiv:1703.04421*, 2017.

[23] J. Alakuijala, R. Obryk, Z. Szabadka, and J. Wassenberg, "Users prefer guetzli jpeg over same-sized libjpeg," *arXiv preprint arXiv:1703.04416*, 2017.

[24] J. Sneyers, E. Ben Baruch, and Y. Vaxman, "CID22: Large-scale subjective quality assessment for high fidelity image compression," *IEEE MultiMedia*, 2023.

[25] Netflix, "Vmaf: Perceptual video quality assessment based on multi-method fusion," https://github.com/Netflix/vmaf.

[26] M. J. Weinberger, G. Seroussi, and G. Sapiro, "The loco-i lossless image compression algorithm: Principles and standardization into jpeg-ls," *IEEE Transactions on Image Processing*, vol. 9, no. 8, pp. 1309–1324, 2000.

[27] J. Alakuijala, R. Van Asseldonk, S. Boukortt, M. Bruse, I.-M. Comșa, M. Firsching, T. Fischbacher, E. Kliuchnikov, S. Gomez, R. Obryk *et al.*, "Jpeg xl next-generation image compression architecture and coding tools," in *Applications of digital image processing XLII*, vol. 11137, 2019, pp. 112–124.

[28] N. Ahmed, T. Natarajan, and K. R. Rao, "Discrete cosine transform," *IEEE Transactions on Computers*, vol. 100, no. 1, pp. 90–93, 1974.

[29] G. Strang, "The discrete cosine transform," *SIAM Review*, vol. 41, no. 1, pp. 135–147, 1999.

[30] C. Loeffler, A. Ligtenberg, and G. S. Moschytz, "Practical fast 1-d dct algorithms with 11 multiplications," in *Proc. IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 1989.

[31] Y. Arai, T. Agui, and M. Nakajima, "A fast dct-sq scheme for images," *IEICE Transactions*, vol. E-71, no. 11, pp. 1095–1097, 1988.

[32] J. Ascenso, P. Akyazi, F. Pereira, and T. Ebrahimi, "Performance evaluation of learning-based image coding solutions and quality metrics," ISO/IEC JTC 1/SC 29/WG 1 (JPEG), 85th JPEG Meeting, San Jose, USA, Technical Report N85013, November 2019. [Online]. Available: https://jpeg.org/jpegai/documentation.html

[33] Y. Maeda, N. Fukushima, and H. Matsuo, "Taxonomy of vectorization patterns of programming for fir image filters using kernel subsampling and new one," *Applied Sciences*, vol. 8, no. 8, 2018.

[34] T. Sasaki, N. Fukushima, Y. Maeda, K. Sugimoto, and S. Kamata, "Constant-time gaussian filtering for acceleration of structure similarity," in *Proceedings of International Conference on Image Processing and Robotics (ICIPRoB)*, 2020.

[35] Y. Kanetaka, S. Asamura, A. Hasegawa, Y. Maeda, and N. Fukushima, "Fast implementation of ssimulacra2 for image quality assessment," in *Proc. International Workshop on Advanced Image Technology (IWAIT)*, 2026.

[36] H. R. Sheikh, M. F. Sabir, and A. C. Bovik, "A statistical evaluation of recent full reference image quality assessment algorithms," *IEEE Transactions on Image Processing*, vol. 15, no. 11, pp. 3440–3451, 2006.

[37] N. Ponomarenko, V. Lukin, A. Zelensky, K. Egiazarian, M. Carli, and F. Battisti, "Tid2008 - a database for evaluation of full-reference visual quality assessment metrics," *Advances of Modern Radioelectronics*, vol. 10, pp. 30–45, 2009.

[38] N. Ponomarenko, L. Jin, O. Ieremeiev, V. Lukin, K. Egiazarian, J. Astola, B. Vozel, K. Chehdi, M. Carli, F. Battisti, and C.-C. J. Kuo, "Image database tid2013: Peculiarities, results and perspectives," *Signal Processing: Image Communication*, vol. 30, p. 57 – 77, 2015.

[39] S. Honda, Y. Maeda, and N. Fukushima, "Dataset of subjective assessment for visually near-lossless image coding based on just noticeable difference," in *Proceedings of International Conference on Quality of Multimedia Experience (QoMEX)*, 2023, pp. 236–239.

[40] S. Honda, Y. Maeda, O. Watanabe, and N. Fukushima, "A multi-level patch dataset for jpeg image quality assessment by absolute binary decision," *IEEE Open Journal of Signal Processing*, vol. 6, pp. 631–640, 2025.

[41] E. C. Larson and D. M. Chandler, "Most apparent distortion: Full-reference image quality assessment and the role of strategy," *Journal of Electronic Imaging*, vol. 19, p. 011006, Jan. 2010.

[42] X. Liu, M. Pedersen, and J. Y. Hardeberg, "Cid: Iq–a new image quality database," in *Proc. International Conference on Image and Signal Processing (ICISP)*, 2014, pp. 193–202.

[43] H. Lin, V. Hosu, and D. Saupe, "Kadid-10k: A large-scale artificially distorted iqa database," in *Proc. of International Conference on Quality of Multimedia Experience (QoMEX)*, 2019, pp. 1–3.

[44] J.-B. Huang, A. Singh, and N. Ahuja, "Single image super-resolution from transformed self-exemplars," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015, pp. 5197–5206.

[45] Y. Matsui, K. Ito, Y. Aramaki, A. Fujimoto, T. Ogawa, T. Yamasaki, and K. Aizawa, "Sketch-based manga retrieval using manga109 dataset," *Multimedia tools and applications*, vol. 76, no. 20, pp. 21 811–21 838, 2017.

[46] L. Jin, J. Y. Lin, S. Hu, H. Wang, P. Wang, I. Katsavounidis, A. Aaron, and C.-C. J. Kuo, "Statistical study on perceived jpeg image quality via mcl-jci dataset construction and analysis," in *Proc. IS&T International Symposium on Electronic Imaging: Image Quality and System Performance XIII*, 2016, pp. 1–9.

[47] G. Bjontegaard, "Calculation of average psnr differences between rd-curves," *ITU SG16 Doc. VCEG-M33*, 2001.