

RESEARCH ARTICLE

Noise-Aware Deterministic Causal Denoising in Predictive Lossless Image Compression

ROMAN STAROSOLSKI^{ID}

Department of Algorithmics and Software, Silesian University of Technology, 44-100 Gliwice, Poland

e-mail: rstarosolski@polsl.pl

This work was supported by Silesian University of Technology under Grant 02080/BK26/0063.

ABSTRACT Lossless compression of noisy images is difficult because noise weakens local dependencies exploited by predictive decorrelation. We investigate whether reversible denoising and lifting steps (RDLS) can improve lossless predictive coding. Building on the lifting interpretation of prediction-error computation, we define an RDLS-modified Median Edge Detector (MED) predictor whose arguments are replaced by deterministic causal denoised estimates while exact reconstruction is preserved. Evaluation uses scalar 8-bit components with synthetic AWGN, real acquisition noise, and a reversible color transform. We measure residual memoryless entropy and EBCOT-coded bit rate. Practical models select one filter per image based on the estimated global noise-level descriptor σ_{milli} , with exhaustive-search ideal selectors as oracle references. On the AWGN-based dataset, ideal selectors in the proposed FAST and VFAST filter spaces exceed 5% EBCOT-coded bit-rate gain, indicating substantial attainable RDLS gain. Practical σ_{milli} -driven regression models retain much of this effect: in LOGO-CV on the same dataset, they reduce average EBCOT-coded bit rate by 0.247 and 0.228 bpp for FAST and VFAST, respectively; in the highest estimated-noise range, both reduce it by 0.520 bpp. Further experiments show that the global noise descriptor is useful but incomplete: block-adaptive VFAST improves over global regression; RDLS remains effective on transformed color components; and RDLS-related memory-to-memory VFAST kernel times are low relative to file-to-file EBCOT times. RDLS raises the MED+EBCOT baseline to rates competitive with JPEG 2000, JPEG-LS, and JPEG XL effort 4 in medium-to-high noise ranges, although higher-effort JPEG XL still gives lower average rates.

INDEX TERMS Deterministic causal denoising, embedded block coding with optimized truncation (EBCOT), lossless image compression, median edge detector (MED) predictor, noise-aware image coding, predictive coding, reversible denoising and lifting step (RDLS).

I. INTRODUCTION

A. LOSSLESS IMAGE COMPRESSION AND NOISY IMAGES

Lossless image compression is important wherever exact reconstruction of image data is required. Examples include medical imaging, scientific imaging, archiving, and acquisition pipelines in which information should not be discarded at an early stage. In such cases, even moderate improvements in the compression ratio may be worthwhile.

A well-known difficulty is that the efficiency of lossless compression depends heavily on the statistical structure of the image. In particular, predictive and transform-based

decorrelation methods are effective mainly because neighboring samples, or transform coefficients, remain statistically dependent. However, this dependence is weakened by noise. As the noise level increases, the benefit of decorrelation may decrease, and in unfavorable cases a decorrelating transform may propagate noise so much that the coding loss caused by noise propagation outweighs the coding gain from correlation reduction. This problem is especially important for images acquired under nonideal conditions and for datasets or acquisition settings with substantially different noise levels.

From the practical point of view, conventional reversible image coding methods may therefore be suboptimal for noisy images not because their coding backend is weak, but because the preprocessing or decorrelation stage is not adapted to

The associate editor coordinating the review of this manuscript and approving it for publication was Jiachen Yang^{ID}.

the noise present in the data. This observation motivates the search for reversible processing methods that are explicitly noise-aware.

B. NOISE-AWARE REVERSIBLE PROCESSING

A natural way of making reversible image coding more robust to noise is to incorporate denoising into the reversible processing chain. The idea is not to denoise the image as a preprocessing step in the usual irreversible sense, but to use denoised quantities inside a reversible transformation. The transformation can then become less sensitive to noise while the original data remain exactly recoverable.

In our earlier work, we proposed reversible denoising and lifting steps (RDLS), originally intended for lifting-based transforms [1], [2]. In RDLS, the raw quantities used in a lifting step (LS), except for the quantity modified by that LS, are replaced by denoised versions computed in a way that keeps the overall transformation exactly reversible. Earlier work showed that such a modification may improve reversible transforms for noisy and some other images, both for 2-D and 3-D reversible discrete wavelet transforms (DWTs) [1], [3], [4] and for reversible color space transforms [2], [5].

C. RDLS-MODIFIED PREDICTION

In the present paper, we investigate whether the same RDLS idea may be applied to predictive coding. More precisely, we build on the interpretation of prediction-error computation as a lifting step and apply RDLS to it by deterministic causal denoising of predictor arguments. In this sense, the paper extends the RDLS design principle from transform coding to a lossless predictive-coding setting. In particular, we apply RDLS to the Median Edge Detector (MED) predictor, which is simple, well known, and representative of practical predictive coding. It is therefore a natural starting point for studying RDLS-modified prediction.

The main idea is straightforward. Instead of using the raw neighboring pixels required by the predictor, we replace them by corresponding estimates denoised in a deterministic and causal way. The predictor itself remains unchanged, except that it operates on denoised arguments rather than on the original ones. Because these denoised values are computed only from causal samples available both during encoding and during decoding, exact reversibility is preserved and the method remains suitable for lossless coding.

Although the present study is restricted to MED, we do not treat the method merely as a special-purpose modification of one predictor. Rather, we view it as a first step toward a broader class of noise-aware RDLS-modified predictive schemes.

D. SCOPE OF THIS WORK

We focus on a practical variant of the problem. The study is limited to scalar 8-bit image components, including grayscale images and individual components of color representations,

and to causal deterministic integer filter implementations. Explicitly specified rounding, clipping, border handling, and operation order ensure bit-exact encoder–decoder agreement. Denoising is not treated as a standalone visual-restoration stage, but as an internal predictor-side operation in a reversible lossless coding loop. Filtered values must be reproducible from samples available identically at the encoder and decoder in raster-scan reconstruction order; exact reversibility, causality, and deterministic execution are therefore fundamental requirements, while practical selection additionally requires compact signaling of the chosen configuration. The objective is not to compare restoration quality across denoisers, but to compare decoder-reproducible predictor-side filtering mechanisms under a common reversible-coding protocol. We investigate three filter spaces of different complexity: FULL, which includes a high-complexity classical nonlocal means (NLM) filter and serves as a reference for the practical spaces, and FAST and VFAST, which are restricted to substantially lower- and very-low-complexity filters, respectively.

The adaptive part of the method uses a fast and simple scalar estimate of the image noise level. Based on this estimate, we build models that select the denoising filter and its parameters. We consider simple lookup models, piecewise models, and regression models, and we assess them using both entropy-based and actual compression-rate-based criteria obtained using a practical entropy coder.

The experiments use two types of noisy data. The first consists of standard natural grayscale images contaminated by synthetic additive white Gaussian noise (AWGN) and is used for model building and internal cross-validation. The second consists of photographic images with different levels of acquisition noise caused by different ISO settings and is used as an additional test of model transferability and of effectiveness beyond the synthetic-noise setting. A compact additional validation also checks components produced by a modular reversible color transform.

E. MAIN CONTRIBUTIONS

The main contributions of this paper are as follows:

- 1) Building on the interpretation of prediction-error computation as a lifting step, we define RDLS-modified prediction for the MED predictor by replacing its causal arguments with deterministic, causally denoised estimates while preserving exact decoder-side reconstruction.
- 2) We construct and evaluate a structurally diverse set of explicitly defined deterministic integer causal filters that can be inserted into RDLS-MED without altering the MED decision rule, including simple filters previously used in RDLS-related studies, local edge-preserving smoothers, local adaptive filters, and a high-complexity classical nonlocal reference; we also characterize the effects of causal support and

- deterministic integer arithmetic relative to the corresponding noncausal or floating-point implementations.
- 3) We introduce three nested filter spaces, denoted FULL, FAST, and VFAST, reflecting different complexity constraints.
 - 4) We propose and investigate adaptive filter selection using lookup, piecewise, and regression models driven by the estimated global noise level, and we add a preliminary block-adaptive VFAST selection variant based on local residual entropy.
 - 5) We evaluate the method on datasets containing images with both synthetic and real acquisition noise, using ideal, exhaustive-search-based per-image selection as an oracle reference, and add a compact validation on components produced by a modular reversible color transform.
 - 6) We compare the resulting RDLS-MED+EBCOT schemes with JPEG-LS, JPEG 2000, and several JPEG XL effort settings, and report implementation-level runtime measurements for the practical VFAST path.

F. ORGANIZATION OF THE PAPER

The remainder of the paper is organized as follows. Section II discusses related work and presents the technical preliminaries needed for the proposed approach. Section III presents RDLS-modified prediction for MED together with deterministic integer and causal filter variants, the adaptive filter-selection framework, and the considered model families. Section IV describes the experimental design, including parameter ranges, datasets, preprocessing, the global noise-level descriptor σ_{milli} , and the evaluation protocol. Section V presents and discusses the experimental results: it first characterizes deterministic integer and causal filter variants relative to their floating-point or full-window references, then evaluates the attainable and model-based coding gains of RDLS-modified MED prediction over unmodified MED prediction, analyzes σ_{milli} -dependent filter-selection behavior, reports a preliminary block-adaptive VFAST selection experiment, includes a compact color-component validation, compares the resulting RDLS-MED+EBCOT schemes with selected standard lossless image codecs, and reports implementation-level runtime measurements for the practical VFAST path. Section VI concludes the paper and outlines directions for further work.

II. RELATED WORK AND PRELIMINARIES

A. LIFTING STEP

Many reversible transforms used in lossless image compression are built from lifting steps (LSs); examples include reversible discrete wavelet transforms and reversible color space transforms [6], [7], [8]. An LS may be defined as

$$C_x \leftarrow C_x \oplus f(C_1, \dots, C_{x-1}, C_{x+1}, \dots, C_n), \quad (1)$$

where C_i is the i th component of the signal being transformed (e.g., a pixel of a grayscale image subjected to a DWT or

a color component of a pixel subjected to a color space transform), C_x is the component modified by the step, and n is the number of components in the signal. The step is reversible if f is deterministic and the operation $\oplus$ is reversible. For DWTs and color space transforms used in lossless image compression, $\oplus$ is addition or subtraction, and f is a weighted sum with truncation or rounding to the nearest integer. A transform built from LSs, or factorized into a sequence of LSs, can be computed in place, is reversible, and has an inverse transform easily constructed from the forward transform. The LS representation also clearly shows that noise may be propagated to the C_x component from other components used in the step.

B. REVERSIBLE DENOISING AND LIFTING STEP

To limit such noise propagation, RDLS was introduced as a modification of lifting steps in which the arguments of the lifting function are denoised before the step is evaluated [1], [2]:

$$C_x \leftarrow C_x \oplus f(C_1^d, \dots, C_{x-1}^d, C_{x+1}^d, \dots, C_n^d), \quad (2)$$

where C_i^d is the denoised version of C_i . For RDLS to be reversible, f must be deterministic and $\oplus$ must be reversible, as for a regular LS. In addition, the denoising itself must be deterministic, and the arguments of f may be denoised using any available component except C_x . Components outside the set used by the original LS may also be used. For example, in a color space transform, denoising may use not only other components of the same pixel, but also components of other pixels in the image. Denoising in RDLS is not an in-place operation: computing C_i^d does not alter C_i , because denoising is performed only temporarily. Thus, RDLS is an LS modification that avoids unwanted noise propagation; unlike denoising applied directly in the image-processing pipeline, it is reversible and therefore suitable for lossless compression.

An RDLS-modified transform retains the main purpose of the original lifting-based transform and preserves its key advantages: it may be computed in place, it is reversible, and its inverse is straightforward to construct. To obtain the inverse of an RDLS-modified transform, we may simply apply inverses of RDLSs in the inverse order. The inverse of the RDLS in (2) is

$$C_x \leftarrow C_x \ominus f(C_1^d, \dots, C_{x-1}^d, C_{x+1}^d, \dots, C_n^d), \quad (3)$$

where $\ominus$ denotes the operation inverse to $\oplus$, so that $c = a \oplus b \Leftrightarrow a = c \ominus b$.

At the same time, the RDLS-modified transform is more general than the original transform, which becomes its special case. The method also allows adaptive filter selection and, in principle, may be applied to any lifting-based transform, not only to transforms used in lossless image compression.

Two special filter options used in RDLS are worth mentioning, although neither is actually a denoising filter: the NONE filter, which leaves the component being denoised

unchanged, and the NULL filter, which replaces every component with 0 [2], [5]. The NONE filter turns RDLS into a regular LS if it is applied to all arguments of f ; see (2) and (1). The NULL filter, if applied to all arguments of f , may result in step skipping for typical lifting steps. In this way, even the entire transform may be skipped. Thus, NULL provides an additional adaptation mechanism when the overall effects of the RDLS-modified transform, including both unwanted side effects and normally desirable decorrelation effects, would worsen the compression ratio.

So far, using only simple denoising filters, we have obtained positive and practically useful effects for several color space transforms [2], [5]. Substantial improvements in compression ratio were also obtained for much more complex multilevel multidimensional DWTs, for nonphotographic images including screen-content images [3], [4]. The latter result is especially interesting because RDLS was effective also for noise-free images, that is, beyond the problem setting for which it was originally introduced.

C. OTHER RELATED APPROACHES

Recent image denoising has developed far beyond local smoothing and classical nonlocal filtering. A recent survey and representative works span classical model-based, supervised CNN, blind/self-supervised, and transformer-based restoration methods, including BM3D, DnCNN, CBD-Net, Noise2Self, Blind2Unblind, MM-BSN, Restormer, and context-aware transformer denoising [9], [10], [11], [12], [13], [14], [15], [16], [17]. Real-noise benchmarks such as DND and SIDD show that synthetic AWGN alone does not characterize denoising on real photographs [18], [19]. These works provide visual-denoising context, but they target full-image restoration and visual metrics, whereas RDLS uses denoising only to produce deterministic causal predictor-side quantities inside a reversible lossless transform.

The following coding-side denoising studies are closer to the present lossless-prediction setting, but they still do not provide direct RDLS-MED baselines. Crandall and Bilgin adapted block matching and three-dimensional collaborative filtering to a causal, decoder-synchronized nonlocal predictor for lossless still-image coding; Wige et al. applied in-loop denoising to temporal reference frames for lossless coding of noisy image sequences [20], [21]. They show that decoder-synchronized denoising-inspired processing is possible, but neither approach provides a drop-in RDLS-MED filter: the former changes the prediction architecture and the latter modifies inter-frame reference data. Here, MED is retained and only its raster-causal arguments are deterministically filtered.

Beyond the RDLS applications summarized in Section II-B, a closely related later line of work inserted denoising into reversible lifting structures for motion-compensated temporal transforms, including Wavelet Lifting with Denoised Update (WLDU), Wavelet Lifting with Denoised Prediction and Update (WLDPU), and a rate-distortion-optimized

variant [22], [23], [24]. These methods preserve exact reconstruction and address noise propagation in temporal lifting for dynamic medical data.

Another related line of work concerns nonlinear and robust predictors for lossless image compression. Early examples include fuzzy switched DPCM predictors for noisy images [25], Boolean and stack-filter predictors used in the prediction stage [26], and soft median adaptive predictors [27]. These methods are relevant because they reduce the influence of noisy or outlying neighboring samples by embedding median-like, fuzzy, or filter-based behavior directly into the prediction rule. The present work follows a complementary route: the MED rule is kept fixed, while the causal values supplied to it are modified by deterministic denoising. More general adaptive prediction methods also train or select predictors from local causal image context to improve decorrelation, but they are usually aimed at adapting to local image structure rather than explicitly compensating noise [28], [29].

Related predictor-side filtering also appears in block-based intra coding. In HEVC and VVC, block-boundary reference samples used for intra prediction may be filtered or smoothed before prediction, typically by simple one-dimensional operations along the available reference row or column [30], [31], [32], [33]. In addition, separate HEVC lossless intra-coding extensions based on 3-tap filters perform pixel-wise prediction using weighted combinations of three already reconstructed neighboring reference samples, where the selected samples and weights depend on the intra mode [34]. Although designed as block-based video-coding tools or HEVC-specific predictor extensions, these methods are conceptually close to the present work because they modify the information supplied to the predictor.

A further related direction is sensor-specific noise compensation inside the predictive loop. Liu et al. proposed a lossless compression method for line-scan infrared push-broom images affected by stripe noise, where pixel-adaptive noise compensation is integrated into a causal five-pixel prediction context, followed by residual hierarchical decomposition and adaptive Golomb–Rice coding [35]. Like the present approach, it avoids separate denoising before compression. Instead, it improves prediction by correcting the context used inside a lossless coder. However, it is tailored to a specific structured noise model caused by detector nonuniformity in line-scan infrared imaging, whereas the present paper studies a more general RDLS-based mechanism for denoising predictor arguments in spatial lossless image prediction.

D. PREDICTIVE CODING

Prediction in lossless coding is defined as follows. Let X_t denote the current image sample. Image samples are processed and indexed in raster-scan order. Let N_t denote the causal neighborhood of X_t , i.e., the set of samples $X_1, \dots, X_{t-1}$ that are both available at the moment of coding X_t and will be available on the decoder side during

reconstruction of X_t . Let $\text{pred}(N_t)$ be a deterministic predictor function of this neighborhood. The task of $\text{pred}(N_t)$ is to predict X_t . Then, in the forward coding stage, the residual sample R_t is computed as

$$R_t = X_t - \text{pred}(N_t). \quad (4)$$

The sequence of residuals, termed the residuum in this paper, is then coded instead of the original image samples; it usually has more favorable statistical properties than the image, e.g., lower entropy, and is therefore more readily compressed.

At the decoder side, reconstruction is performed using the same predictor applied to the already reconstructed causal neighborhood N_t . The original sample is recovered as

$$X_t = R_t + \text{pred}(N_t). \quad (5)$$

Predictive coding may therefore be viewed as a reversible operation that removes correlation between a pixel and its causal neighborhood. Although predictive coding is usually presented in the classical form given above, Section III-A shows that it may also be reformulated as a lifting-based reversible transform, making the application of RDLS to it straightforward.

E. THE MED PREDICTOR

In the present study, we use the Median Edge Detector (MED) predictor, also known as the LOCO-I predictor and standardized in JPEG-LS [36], [37]. It is based on three causal neighbors, usually denoted by A , B , and C , corresponding respectively to the left, upper, and upper-left samples. The predictor may be written as

$$\text{MED}(A, B, C) = \begin{cases} \min(A, B), & \text{if } C \geq \max(A, B), \\ \max(A, B), & \text{if } C \leq \min(A, B), \\ A + B - C, & \text{otherwise.} \end{cases} \quad (6)$$

Equation (6) shows the nonlinearity of MED and its ability to handle simple horizontal and vertical edge configurations. It can also be written as

$$\text{MED}(A, B, C) = \text{median}(A, B, A + B - C), \quad (7)$$

which explains the term Median Edge Detector and relates it to median-adaptive prediction. The MED predictor is well established in predictive lossless coding because it is simple, fixed, and fast, while still adapting to the most important local edge configurations. More elaborate predictors, such as the Gradient-Adjusted Predictor (GAP) used in CALIC, employ a larger causal neighborhood and gradient-dependent decision rules to better adapt prediction to local image structure [38]. MED is therefore a meaningful test case for the present study: it is simpler and more standardized than GAP, but still nonlinear and representative of practical predictive lossless coding.

III. PROPOSED METHOD

A. PREDICTION AS A LIFTING TRANSFORM AND ITS RDLS-MODIFIED VERSION

We start from the reconstruction of image samples from residual samples. On the decoder side, see (5), once a sample X_t has been reconstructed from a residual sample R_t , R_t is no longer needed. Only X_t and the image samples reconstructed earlier from residual samples are needed, both because they form the causal neighborhood for reconstructing subsequent samples and because they are part of the final reconstructed image. Therefore, instead of (5), we can use the following formula for in-place computation of image samples:

$$C_t \leftarrow C_t + \text{pred}(N_t), \quad (8)$$

where C_t is a residual sample R_t being modified to become the image sample X_t . After applying (8) in raster-scan order to all residual samples, we obtain the original image in place of the residuum. At the same time, (8) is an LS; compare it with (1). Therefore, RDLS may be directly applied to (8) by replacing the predictor arguments N_t with their denoised versions N_t^d , yielding the RDLS-modified in-place reconstruction

$$C_t \leftarrow C_t + \text{pred}(N_t^d). \quad (9)$$

However, when denoising N_t , we should use only the already reconstructed samples, that is, N_t itself, which corresponds to $C_1, \dots, C_{t-1}$ in the notation of (8). All other samples, including C_t , are still residual samples when (9) is applied.

Since reconstruction is a sequence of LSs, prediction may also be expressed in this form by applying inverse LSs in the inverse order. The same relation holds between RDLS-modified reconstruction and RDLS-modified prediction; recall (2) and (3). Thus, to obtain RDLS-modified prediction, we apply the inverse of (9),

$$C_t \leftarrow C_t - \text{pred}(N_t^d), \quad (10)$$

in the inverse order, i.e., inverse raster-scan order. Naturally, because we use the same $\text{pred}(N_t^d)$ as in (9), the samples in the causal neighborhood and the denoising method are the same.

We also note that, both for LS-based prediction and for RDLS-modified prediction, the predicted sample is computed only from original image samples. Therefore, we may return to a more implementation-friendly, not-in-place version that reads image samples and computes RDLS-modified prediction residual samples as new values:

$$R_t = X_t - \text{pred}(N_t^d). \quad (11)$$

An important advantage of (11) over (10) is that it may be applied to image samples in any order, in particular in raster-scan order, which is usually the most suitable for implementations. Although prediction using (11) may have access to all samples of the entire image, denoising must still use only N_t , because only these image samples will be available during reconstruction.

Thus, to apply RDLS to MED prediction, we use the same MED rule as without RDLS, but replace the causal neighbors

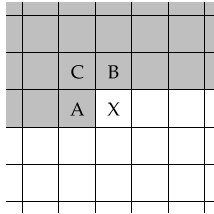


FIGURE 1. Causal neighborhood, marked in gray, of the predicted pixel X and locations of the three nearest neighbors used by MED: A , B , and C .

A , B , and C by their corresponding causally denoised estimates A^d , B^d , and C^d ; briefly, we use $MED(A^d, B^d, C^d)$. We refer to this RDLS-modified MED prediction method as RDLS-PredMED.

B. DETERMINISTIC CAUSAL DENOISING; CROPPED, REPLICATE, FILTERING-A, AND FILTERING-ABC

In this study, we use the MED predictor together with several classic simple and advanced denoising filters. The application of denoising in RDLS-modified prediction imposes additional restrictions on how the filtering method may be applied.

Some filters are normally implemented using floating-point arithmetic, which generally does not guarantee bit-exact results when the filtering program is compiled with different compilers or options, or executed on different hardware. For this reason, the research implementation uses deterministic integer or fixed-point definitions for all experimental filters.

Real-valued filter parameters are represented deterministically as integers where applicable, typically in milli-units for externally specified parameter values. Intermediate computations use filter-specific integer or fixed-point arithmetic. The exact scaling, rounding, clamping, lookup-table, and tie-breaking rules are documented in the accompanying application reference. Section V-A evaluates the closeness of these definitions to the corresponding floating-point implementations.

Most filters considered here are based on square local windows, also described as filter kernels or filter supports, centered on the pixel being denoised. The main exception is the directional median filter, which uses directional subsets; these may be viewed as a square window in which certain sample locations are assigned zero weight or ignored. The NLM filter also relies on square neighborhoods, but distinguishes between two different windows: a larger square search window and a smaller square patch window, whose center is constrained to lie inside the search window. However, for denoising of the causal neighborhood, the lower part of the filter window, up to almost half of the window, is placed outside the causal area and must be excluded from computations; see Fig. 1. Below, we discuss the implications for RDLS-modified prediction.

The actual number of pixels in the causal window is smaller than the squared side length of the full filter window. This may be problematic for small windows, especially because

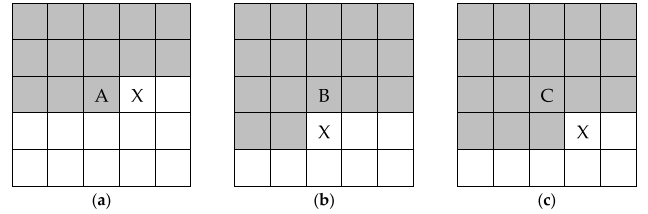


FIGURE 2. Causal pixels, marked in gray, inside a 5×5 filter window for denoising of MED neighbors A (a), B (b), and C (c) of the predicted pixel X .

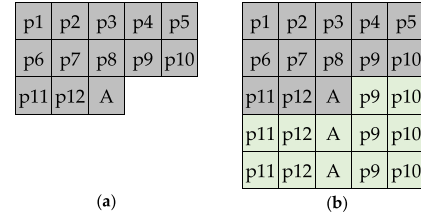


FIGURE 3. Actual window available for filtering in the cropped (a) and replicate (b) causal window modes. The example shows denoising of neighbor A with the original 5×5 filter window; see Fig. 2(a). Actual causal pixels are marked in gray, and copies of causal pixels used to fill the virtual full-size window in the replicate causal mode are marked in light green.

some filters reject selected window samples. One way to alleviate this limitation is to use larger windows.

The shape of the causal area of the filter window differs for different predictor arguments. Fig. 2 shows these areas for the MED predictor when a 5×5 filter window is used. The size difference is especially important for the smallest windows. For example, for a 3×3 window, the causal area for pixel C contains eight pixels, whereas the causal area for pixel A contains only five pixels. We may denoise each predictor argument using its own causal-area shape, containing all samples causally available for this argument. This approach is expected to give the best denoising, but it may require filtering the same image pixel up to three times, once for each of its possible roles as neighbor A , B , or C in the prediction of another pixel. We denote this method by *filtering-abc*. Alternatively, all predictor arguments may be denoised using the common part of their causal areas, which for MED is the shape for neighbor A ; see Fig. 2(a). Then each image pixel may be denoised once and reused several times, either by caching the denoised pixel or by denoising the entire image beforehand and using it as a temporary source of denoised samples. However, some predictor arguments are then not denoised using the entire causal area available for them. We call this variant *filtering-a*.

Some filters ignore the relative positions of filter-window pixels with respect to the pixel being denoised. For example, the box filter averages all pixels in the window, which may result in biased estimates. If the window contains a vertical gradient, the filtered value will be biased with respect to the central pixel even in the absence of noise. Other filters, such as the bilateral filter, do consider sample positions. Taking into account the properties of both types of filters, we define two causal window modes: *cropped* and *replicate*. In the *cropped* mode, the original square

filter window is cropped to the causal area; see Fig. 3(a). In the `replicate` mode, noncausal samples in the filter window are replaced with their nearest causal neighbors located directly above them; see Fig. 3(b). Compared with `cropped`, the filter then receives a virtual full-size window: pixels above the bottom border of the actual causal area receive less total weight, whereas the bottom border itself receives more weight. For the earlier box-filter example with a vertical gradient, this would reduce, but not remove, the bias. We denote `filtering-abc` combined with `cropped` and `replicate` as `cropped-abc` and `replicate-abc`, respectively; similarly, we denote `filtering-a` combined with these causal modes as `cropped-a` and `replicate-a`, respectively.

C. FILTER FAMILIES AND PRACTICAL FILTER SUBSETS

The filter set used in this study includes both simple and more advanced denoising families. The filter set is not intended as a benchmark of current visual-denoising methods; rather, it provides a structurally diverse RDLS-compatible candidate space. Earlier RDLS applications were investigated mainly with simple filters, namely `NULL`, `NONE`, `box`, and `median` filters, whereas the use of more advanced filters was left for future work. Because the present paper applies RDLS in a different setting, namely predictive lossless coding, it was not evident that the filters useful in earlier RDLS-modified transforms would also be the most suitable here. Therefore, we considered a broader set of candidate filters to assess RDLS-modified MED prediction in a sufficiently rich RDLS-compatible filter space.

Besides the `NULL` and `NONE` cases described in Section II-B, we used a set of denoising filters that may be grouped into several classes. First, we considered basic linear and order-statistics filters: the box filter, the median filter, and the alpha-trimmed mean filter; the latter two belong to the classical nonlinear order-statistics filtering family [39]. We also included more advanced median-based filters: weighted-median filters, including a flat variant and two position-dependent variants with power and exponential distance weighting [40], [41], and the directional median filter [42]. In addition, we used edge-preserving local smoothing filters: the bilateral filter [43], the guided filter, with the image itself used as the guidance signal [44], and the local adaptive Wiener filter [45]. Finally, the NLM filter was included as a high-complexity classical nonlocal reference filter capable of preserving repeated structural patterns [46]. Short characteristics of all these filter families are given in Appendix A-A.

From a practical point of view, the considered filters have substantially different computational costs, even for reasonably optimized 8-bit implementations. Here, we characterize these costs only coarsely, using approximate per-pixel orders with respect to the filter-window dimensions and informal constant-factor descriptions; measured runtime for the practical VFAST path is reported separately in Section V-I.

Unless stated otherwise, the following estimates apply to both the deterministic full-window implementations and their causal `cropped` and `replicate` variants. For most filter families, restricting the support to causal samples changes mainly the constant factor rather than the approximate order; moreover, `filtering-abc` may require up to three filter evaluations per predicted pixel, whereas `filtering-a` requires one. Let $k = (2r + 1)^2$ denote the nominal filter-window size. The box filter may be treated as effectively $\mathcal{O}(1)$, with a very small constant, when running sums or an integral image are used. Median, alpha-trimmed mean, and flat weighted median filters may also be treated as effectively $\mathcal{O}(1)$ for 8-bit data with histogram-based updates, although with larger constants. Position-dependent weighted median variants and the directional median filter are more demanding and may be treated approximately as $\mathcal{O}(r)$, with relatively large constants. Exact bilateral filtering is approximately $\mathcal{O}(k)$, with a large constant because of the range weighting. Conventional full-window guided filtering may be treated as effectively $\mathcal{O}(1)$ per pixel when running local statistics are used. In the causal guided formulation used in this study, the fixed-prefix support prevents this constant-time implementation; with integral-image acceleration, the per-pixel cost is approximately $\mathcal{O}(k)$. The local adaptive Wiener filter remains effectively $\mathcal{O}(1)$ for both full and causal support because it requires only local first- and second-order statistics rather than nested local-window computations. Finally, NLM is by far the most computationally demanding family considered here, with complexity approximately $\mathcal{O}(k_p k_s)$, where k_p and k_s are the patch and search-window sizes, respectively.

Because of these large complexity differences, we define three nested filter spaces, from which the models described in Section III-D adaptively select a denoising filter for a given image. The `FULL` space contains all tested filter families and serves as the broadest reference space showing the coding-rate or entropy gains attainable when accepting complexity levels that are usually impractical. The `FAST` space excludes the computationally most demanding NLM family, while retaining the remaining filters. The `VFAST` space is restricted to the simplest box filter together with the special cases `NONE` and `NULL`, and is intended as the practical, low-complexity, and easy-to-implement variant. The use of these three spaces allows us to assess not only the coding gains attainable with adaptive RDLS-modified MED prediction, but also their dependence on the admissible practical complexity of the filter set.

Accordingly, direct experimental candidates were restricted to filters with fully specified raster-causal support, boundary handling, parameterization, and deterministic integer or fixed-point arithmetic that could be inserted into RDLS-MED without changing the underlying prediction architecture. Recent learned blind-spot, transformer-based, and other full-image or noncausal denoisers remain relevant related work, but their published forms do not define such bit-exact filters for the MED arguments. Including them would require

designing and validating new constrained RDLS variants rather than directly implementing the published methods, which is outside the scope of the present study.

To further characterize the computational cost of the most practical VFAST space, we additionally estimated the elementary per-pixel operation count of optimized causal box variants. Counting basic integer arithmetic, histogram updates, and conditional assignments as one operation and division/normalization as two, the running-sum box filter with unit center weight requires approximately 9, 15, 13, and 22 operations per predicted pixel in *cropped-a*, *cropped-abc*, *replicate-a*, and *replicate-abc*, respectively; nonunit center weights add 2 operations in *filtering-a* and 6 in *filtering-abc*. These estimates are only coarse and implementation-independent: they exclude memory traffic, row-state initialization, border handling, and prefix/running-sum overhead. The measured implementation-level timings, including independent A/B/C computations in *filtering-abc*, are reported in Section V-I.

D. MODELS AND VARIANTS OF ADAPTIVE FILTER SELECTION

For a given image, a single filter configuration is selected for all pixels and predictor arguments based on the scalar noise-level descriptor σ_{milli} , whose computation is defined in Section IV-D. A related image-adaptive filter-selection strategy was previously used for RDLS-modified color component transforms, where the model was driven by acquisition metadata, namely the ISO sensitivity used during image acquisition [2]. In contrast, the descriptor used here is computed directly from the image, so the model does not require acquisition metadata. Its role is not to characterize image noise in full detail, but to provide a practical and robust global control variable for selecting the denoising filter and its parameters. This is sufficient for the present purpose, as the considered models select from a finite set of filter configurations rather than estimate filter parameters directly.

We define three model families. The first family contains one simple lookup model, in which each value of σ_{milli} is assigned the filter configuration that performs best on average for the training-set images having this value of σ_{milli} . For σ_{milli} values not present in the training data, the model selects the filter configuration assigned to the closest value of σ_{milli} present in the training set. This model directly reflects the empirical relation between the estimated noise level and the preferred filter configuration and may therefore be treated as the simplest adaptive baseline; however, it may be sensitive to outliers or sparsely represented σ_{milli} values.

The second family consists of piecewise-constant models, in which the range of σ_{milli} is partitioned into a small number of intervals and one filter configuration is assigned to each interval. In the present study, we consider models with 3, 4, and 5 segments. The piecewise models are trained by minimizing the total empirical coding cost over the training data for a fixed number of segments. For this

purpose, the attainable values of σ_{milli} are ordered, and dynamic programming is used to determine the partition that minimizes the cumulative cost. The cost of a segment is defined as the total empirical coding cost obtained when a single filter configuration is assigned to all samples whose σ_{milli} falls into that segment. In this way, the model finds a compact segmentation of the noise-estimation axis that approximates the best filter-selection rule achievable within the assumed number of segments.

The third family consists of regression models, in which the expected coding gain of each candidate filter configuration is modeled as a function of σ_{milli} , and the configuration with the largest predicted gain is selected. In the present study, we use quadratic regression. Let x denote the regression variable derived from σ_{milli} . We consider the following simple variants:

$$\begin{aligned} x_{\text{raw}} &= \sigma_{\text{milli}}, \\ x_{\text{log}} &= \ln(1 + \sigma_{\text{milli}}), \\ x_{\text{sqr}} &= \sqrt{1 + \sigma_{\text{milli}}}, \end{aligned}$$

denoted as *quad_raw*, *quad_log*, and *quad_sqr*, respectively. For each candidate filter configuration q , the predicted gain is modeled as

$$g_q(x) = a_{0,q} + a_{1,q}x + a_{2,q}x^2.$$

The coefficients $a_{0,q}$, $a_{1,q}$, and $a_{2,q}$ are determined from the training data by least-squares fitting, i.e., by minimizing the sum of squared differences between the empirical gains observed for configuration q and the gains predicted by the quadratic regression model.

The use of transformed versions of σ_{milli} is motivated by the expected shape of the relation between coding gain and noise level. We expected the most dynamic changes to occur for small noise levels, while for a broad range of larger values the dependence would vary more slowly. For this reason, it is natural to consider transformations that stretch the lower part of the range and compress the higher part.

As the regression model is intended mainly for interpolation within the noise-level range represented in the training data, rather than for extrapolation beyond it, the input σ_{milli} is clipped to the training range before the fitted regression function is evaluated. In addition, to reduce the risk of unrealistic overestimation of gain, we apply a simple overshoot protection mechanism: for a given filter configuration, the predicted gain is not allowed to exceed the maximum empirical gain observed for that configuration in the training data.

In addition to the above practical models, we also use an ideal selector that chooses, by exhaustive search and ignoring σ_{milli} , independently for each image, the filter configuration that yields the best result according to the criterion being optimized. Its role is to provide an oracle reference for the gain attainable within a given predefined filter space and thereby to assess the efficiency of the practical models.

All model families are considered separately for the FULL, FAST, and VFAST filter spaces introduced earlier. They are also built separately for the optimization target, namely either the empirical memoryless entropy H_0 of the residuum or the actual compression rate c_r obtained by entropy coding of the residuum. The details of model training, validation, and evaluation protocol are given in Section IV.

As a preliminary test of local adaptivity, we additionally evaluated block-adaptive VFAST selection. The image was divided into square blocks, and one VFAST configuration was selected for all pixels predicted inside a block by minimizing the local memoryless entropy of the RDLS-MED residual in that block. The causal filtering support was not reset at block boundaries. The VFAST dictionary contained 254 candidates: NULL, NONE, and box configurations with the same radii, center weights, causal modes, and filtering strategies as in the global VFAST experiments. A fixed one-byte configuration index was added for every block; the corresponding side-information size was included in all reported H_0 and c_r values.

IV. EXPERIMENTAL DESIGN

A. REPORTED MEASURES

In this study, we aim to improve the compression rate of a predictive coding method. To evaluate this effect, we use two measures of the compressibility of the residuum, or of the unprocessed image when no prediction is applied. The first measure is the empirical memoryless entropy H_0 , which provides a general coder-independent estimate of compressibility. The second measure is the actual compression rate c_r obtained with a practical coder, namely the Embedded Block Coding with Optimized Truncation (EBCOT) context arithmetic coder from the JPEG 2000 standard [47]. Both measures are expressed in bits per sample, equivalently in bits per pixel (bpp) of the original image; therefore, smaller values mean better compression.

For a sequence of samples, we compute the memoryless entropy as

$$H_0 = - \sum_{i=S_{\min}}^{S_{\max}} p_i \log_2 p_i, \quad (12)$$

where $S_{\min}$ and $S_{\max}$ are the smallest and largest sample values in the residuum, or the smallest and largest pixel values of the original component if H_0 is applied directly to the component. The value p_i denotes the empirical probability of the sample value i . We assume the usual convention $0 \log_2 0 = 0$ and express H_0 in bpp.

For the EBCOT-based coding experiments, we use the Kakadu implementation of JPEG 2000; the implementation details and command-line settings are given later in the experimental-design section. The compression rate, also termed the bit rate, is computed from the total size s in bytes of the compressed residuum:

$$c_r = \frac{8s}{M}, \quad (13)$$

where M is the number of samples in the residuum. For all codecs reported in this paper, the bit rate is computed from the total compressed file length, including the compressed file format and/or container header. For EBCOT coding of predictive residuals, the signed residual is stored as a nominal 9-bit residual image. With 8-bit source samples and prediction value P_i clamped to $[0, 255]$, the value written to the residual PGM plane is $Y_i = X_i - P_i + 255$, so $Y_i \in [0, 510]$ and the PGM file is written with `Maxval=511`. The inverse path reconstructs $X_i = Y_i - 255 + P_i$ using the same decoder-reproducible prediction.

Except for the preliminary block-adaptive experiment described below, we do not include three small side-information costs in the reported rates. First, the transmitted one-byte quantity would be the integer median-gradient index from which σ_{milli} is reconstructed by the fixed 256-entry integer table used by the implementation, not the decimal value of σ_{milli} itself. Second, the selected filter, its parameters, filtering mode, and causal mode would require at most two bytes if explicitly signaled, but they need not be signaled when encoder and decoder use the same deterministic model table. Third, the original-image histogram needed to invert offline histogram packing (HP) requires 32 bytes with the bit-array encoding described in Section IV-B. For the image sizes considered in this study, the combined contribution of these items is negligible. For the block-adaptive VFAST experiment, the fixed one-byte-per-block configuration map is included in the reported H_0 and c_r values.

The accompanying RDLS-pred research application reference documents implementation-level conventions that are intentionally not expanded in the manuscript, including MED and causal-filter boundary handling, details of σ_{milli} estimation, model-table selection, metadata sidecars, and block-map semantics.

For either measure $m \in \{H_0, c_r\}$, we compute the coding gain due to RDLS-modified prediction relative to unmodified MED prediction as the percentage improvement

$$G_m = 100 \frac{m_{\text{MED}} - m_{\text{RDLS}}}{m_{\text{MED}}}, \quad (14)$$

where m_{MED} is the value obtained with unmodified MED prediction and m_{RDLS} is the corresponding value obtained with RDLS-modified MED prediction. A positive value of G_m therefore denotes an improvement. Unless explicitly stated otherwise, percentage gains reported in the model and diagnostic tables are arithmetic means of individual per-image or per-component gains, so each image/component has equal weight. Tables that report average bit rates instead average the rates themselves; consequently, a percentage computed from the difference of two average rates need not equal the mean of the individual percentage gains.

For comparing filtering outputs, we use the standard peak signal-to-noise ratio (PSNR) for 8-bit components, computed from the mean squared error with peak value 255; identical outputs are reported as infinite PSNR.

B. HISTOGRAM PACKING

For some images, the number of active levels, i.e., intensity levels actually used by image pixels, may be much smaller than implied by the nominal bit depth. This can happen even when the active levels are distributed over the full nominal intensity range and the image visually appears as if its histogram did not contain many empty bins. Such images have sparse histograms of intensity levels [48].

Histogram sparseness has a detrimental effect on the compression rate of predictive coding because both the coders, including EBCOT, and prediction in general are not designed for sparse-histogram image data. The residuum of a sparse-histogram image is usually not sparse; it may therefore have a greater H_0 and be compressed less efficiently than the original image. To alleviate this problem, we may use the HP technique, which simply maps all active levels to the lowest part of the nominal intensity range using an order-preserving one-to-one mapping.

For example, among the photographic images in the Waterloo Greyscale Set 2 described in the next section, the effect is most pronounced for the `mountain` image, which uses roughly 40% of the pixel intensity levels between the darkest and brightest levels actually present in the image. For this unprocessed image, H_0 and c_r are 6.224 bpp and 6.251 bpp, respectively, and after prediction these measures worsen to 6.805 bpp and 6.545 bpp, respectively. If HP is applied before prediction, then H_0 and c_r after prediction are reduced to 5.625 bpp and 5.432 bpp, respectively, i.e., by over 17%.

HP is performed before compression, and to invert it after decompression the original histogram must be encoded and transmitted to the decoder. In this study, we use the bit-array encoding method [49]; that is, for each of the 256 nominally available levels, we encode one bit of information specifying whether this level is active. Thus, the encoded histogram requires 32 bytes regardless of the number of active levels.

Except where explicitly stated otherwise, namely for one branch of the color-transform validation and for the separate runtime measurement of HP itself, the experiments are performed on histogram-packed scalar components, including grayscale images.

C. DATASETS

In this study, we investigate predictive compression of scalar 8-bit photographic image components, with special attention to noisy photographic data. As the main dataset used for model construction and internal validation, we adapted a well-known set of diverse grayscale photographic images, namely Waterloo Greyscale Set 2 [50]. However, of the 12 images in that set, `library` and `france` are nonphotographic, and the `lena` image was excluded for ethical and publication-policy reasons. Therefore, we used the remaining nine images, shown in Fig. 4(a). These images are diverse, but not very noisy. For each image, we generated variants with added AWGN using `pgmtool` 0.73b [51], with standard deviations $\sigma \in \{2, 3, 5, 7, 10, 15, 20, 30, 50\}$ and clipping the noisy samples to the nominal 8-bit range 0, ..., 255. For

the reasons presented in Section IV-B, we used images with packed histograms, and AWGN was added after applying HP. The wide range of image characteristics and noise levels should allow us to construct models suitable for photographic images with noise levels ranging from low to high. The resulting set contains 90 images, with sizes ranging from 512×512 to 621×498 . We denote this set by `GS2noise`.

The second dataset is used mainly for additional external testing, to assess both model transferability and method effectiveness beyond the synthetic-noise setting. It consists of photographic images affected by varying levels of real acquisition noise rather than synthetic AWGN. We used the A3 set introduced in earlier research [2], [5], available from [52], which contains 116 images acquired using a camera equipped with a Bayer-pattern RGGB color filter array and only minimally processed after RAW acquisition. The Bayer-pattern image was converted to a color image by taking, for each group of RGGB subpixels, the average of the two green subpixels as the G component and the red and blue subpixels as the R and B components of the RGB image, respectively. Among its images, the A3 set contains six series, each depicting a different scene captured with fixed manual focus and aperture but at eight ISO speeds from ISO 100 to ISO 12800. We used only the images from these series; examples are shown in Fig. 4(c) and (d). As the method is applied to scalar components, we treated the R, G, and B components of each image as separate components in the main A3series experiments. Similarly to the `GS2noise` set, we applied HP to these components. In total, this yielded 144 components, all of size 1992×1493 . We denote this set by `A3series`.

To further check the component-level applicability of the method in a color-transform setting, we also formed two additional A3series representations using the modular reversible color-transform example specified in JPEG-LS Part 2, Annex F.2 [53]. The transformed components were generated using `colortransf` 1.1 [51]. This transform was chosen because it belongs to the JPEG-LS extensions family, keeps the validation close to the MED/JPEG-LS predictive-coding setting, and preserves the nominal 8-bit component depth used by the validated implementation. We evaluated two color-transformed component sets: `mRCT`, obtained by applying the modular reversible color transform directly to the RGB components without prior HP, and `HP+mRCT`, obtained by first applying HP independently to the RGB components and then applying the same modular color transform. Both variants were tested because the RGB components, especially noisy R and B components, may have sparse histograms and HP was used in the RGB-component experiments, whereas HP is not a standard prerequisite before a color transform. RDLS-PredMED itself was not modified and was applied independently to each resulting component.

D. NOISE-LEVEL ESTIMATION

In the present study, the σ_{milli} noise-level estimator used by the filter-selection models and by noise-scaled filter

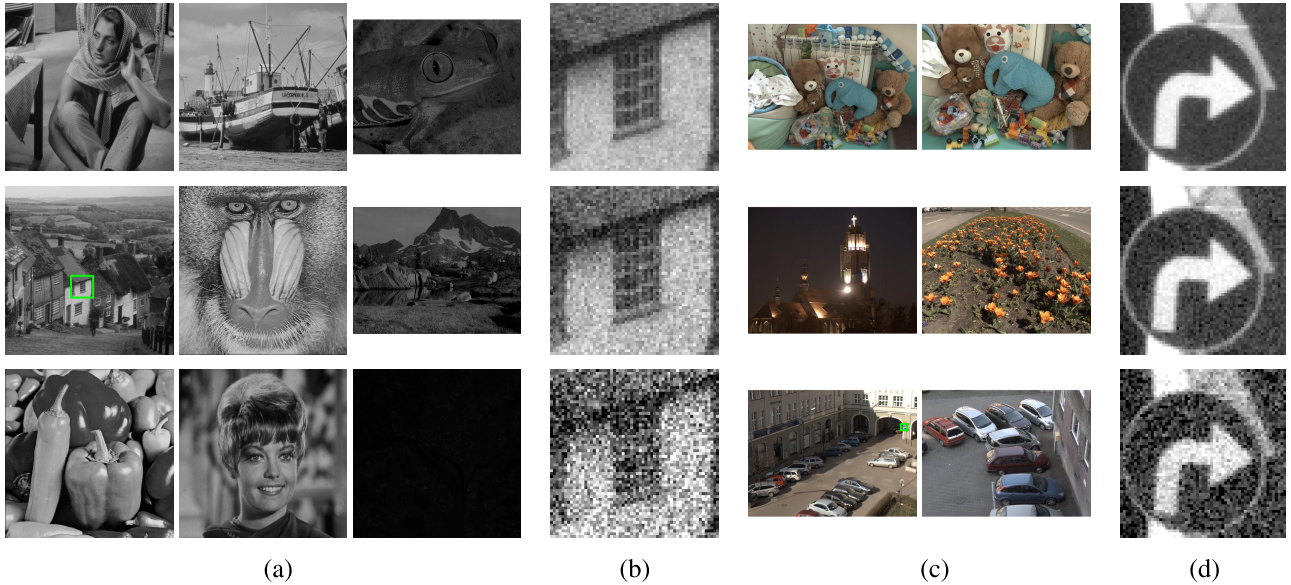


FIGURE 4. Datasets used in the experiments: (a) GS2noise source images after HP and without added AWGN; (b) a crop from one GS2noise image after adding AWGN with $\sigma = 10, 20$, and 50 ; (c) ISO 100 color images from the six A3series scenes, whose R, G, and B components were processed as separate scalar components; and (d) crops of the R component from one A3series scene acquired at selected higher ISO settings, namely ISO 800, ISO 3200, and ISO 12800. The green rectangles in panels (a) and (c) mark the locations of crops shown in panels (b) and (d), respectively.

parameters is obtained from a gradient-based median absolute deviation (MAD) estimator applied to the image before prediction [54]. More specifically, we compute absolute first-order differences between horizontally and vertically neighboring pixels and collect all such nearest-neighbor gradients over the whole image. The median of these values is then multiplied by $1.4826/\sqrt{2}$, which yields a robust estimate of the noise standard deviation under the usual Gaussian-noise assumption for nearest-neighbor differences.

For use in integer-implemented filters, the resulting value is next multiplied by 1000 and rounded to the nearest integer. Thus, σ_{milli} is a fixed-point representation of the estimated noise standard deviation, with 1000 corresponding approximately to 1.0. In the implementation, the median is obtained from a 256-bin histogram of integer nearest-neighbor gradients, and the possible integer median-gradient indices are mapped to σ_{milli} by a fixed table. Therefore, the value used by the models should be regarded as a coarse but practical descriptor of the global noise level rather than as a highly precise estimate. This is sufficient for the present purpose, as the considered models select from a finite set of filter configurations rather than estimate parameters directly.

The computational cost of this estimator is small. For each pixel that is not in the top row and not in the left column, the implementation accumulates two nearest-neighbor gradients, namely the differences to the left and upper samples. Counting subtraction, absolute value, and histogram increment as one operation each, this gives about six elementary integer operations per such pixel. The final scan of the 256-bin gradient histogram to find the median and the subsequent fixed-table lookup are independent of image size and negligible after normalization per pixel. Therefore, computing σ_{milli} , which is performed only once

per image, has an elementary-operation count comparable to, and slightly smaller than, that of MED prediction.

E. MODEL TRAINING AND VALIDATION

The adaptive models described in Section III-D were trained and evaluated separately for GS2noise and A3series. For GS2noise, internal validation was based on leave-one-group-out cross-validation (LOGO-CV), where each group consisted of one original image together with all of its noisy variants. Thus, in each fold, the model was trained on all groups except one and tested on the omitted group. This makes the validation stricter than random splitting, as the model must generalize to an unseen image rather than only to unseen noise levels.

For A3series, the data are organized into six series corresponding to six scenes, each acquired at several different ISO settings. Because two of these series contain very similar image content, one of them being effectively a zoomed fragment of the other scene (see the top of Fig. 4(c)), we did not use the six series directly as separate LOGO-CV groups. Instead, we formed three disjoint groups of equal size, each containing two series, and used these three groups in validation. This reduces the risk of content leakage between training and test folds while keeping the protocol simple and balanced.

In addition to internal validation, we also performed cross-dataset testing, in which models trained on the entire GS2noise set were tested on A3series in order to assess transferability from synthetic-noise data to real acquisition noise.

All model families were trained separately for the FULL, FAST, and VFAST filter spaces, and separately for the optimization targets H_0 and c_r considered in this study. The

TABLE 1. Comparison of deterministic full-window, noncausal implementations of advanced filters with equivalent floating-point OpenCV-based versions. Family-wise PSNR quartiles over all tested configurations on GS2noise.

Filter family	PSNR Q1 [dB]	PSNR median [dB]	PSNR Q3 [dB]
Bilateral	43.95	49.97	54.88
Guided	55.05	61.84	67.45
Wiener	72.82	75.09	79.95
NLM	48.66	54.47	59.99

ideal selector was obtained independently for each image by exhaustive search and was used only as an oracle reference.

F. IMPLEMENTATION DETAILS AND CODEC SETTINGS

The research implementation of the proposed RDLS-modified MED prediction, written in C++17, is provided as supplementary material with this submission. This main research implementation was designed primarily for deterministic and reproducible evaluation rather than for runtime optimization.

For the comparison of deterministic advanced denoising filters with corresponding floating-point references reported later in the results section, we used the OpenCV 4.12.0 implementations invoked from the research implementation [55]. The compression rate of the EBCOT entropy coder [47] for RDLS-modified MED residual images was obtained using a JPEG 2000 implementation with the DWT transform stage switched off. We used Kakadu 8.6.2 [56], invoked with `kdu_compress -i in.pgm -o out.j2k, Creversible=yes, -rate -, and Clevels=0`.

The image compression rates of the standard JPEG-family codecs, namely JPEG-LS [36], [37], JPEG 2000 [57], [58], and JPEG XL [59], [60], were obtained using the following implementations and settings. For JPEG 2000, we used the same Kakadu 8.6.2 implementation, but with the default DWT decomposition, invoked with `kdu_compress -i in.pgm -o out.j2k, Creversible=yes, and -rate -`. For JPEG XL, we used libjxl 0.11.2 in lossless mode, invoked for each tested effort value $e \in \{1, 4, 7, 10\}$ with `cjxl, --distance 0, --effort e, in.pgm, and out.jxl` [61]. For JPEG-LS, we used CharLS 2.4.3 through a small command-line wrapper with default encoder settings [62].

The JPEG 2000 and EBCOT-backend experiments were additionally repeated with OpenJPEG 2.5.4 [63] using corresponding lossless settings. The resulting model summaries differed only negligibly from the Kakadu-based summaries, and the rankings and conclusions were unchanged.

To assess the practical cost of the low-complexity VFAST path, we additionally implemented a separate single-threaded runtime benchmark application, also provided as supplementary material. The benchmark uses performance-oriented memory-to-memory kernels for HP, histogram-based σ_{milli} estimation, H_0 , MED forward and inverse prediction,

NULL/NONE paths, and causal box filtering. The benchmark implementation is performance-oriented but is not intended to be a production codec; in particular, it separates border and interior handling and uses prefix/running sums for causal box operations. No SIMD/vectorization, profile-guided optimization, multithreading, or other hardware-specific low-level tuning was used, and the `filtering-abc` variants were not integrated into a single shared A/B/C kernel. External Kakadu and OpenJPEG codec calls were timed separately as file-to-file operations.

V. RESULTS AND DISCUSSION

A. ACCURACY OF DETERMINISTIC INTEGER AND CAUSAL FILTERING

Before analyzing the effect of causal deterministic denoising of predictor arguments on the entropy and bit rate of prediction residuals, we characterize how the integer and causal variants of filtering differ from the corresponding standard or noncausal implementations. We begin by comparing the integer, but still noncausal, full-window versions of the advanced filters normally implemented using floating-point arithmetic, namely bilateral, guided, Wiener, and NLM filters, with their floating-point counterparts. Recall that in our experiments these filters were implemented using deterministic integer or fixed-point definitions, with filter-specific scaling, rounding, and lookup-table rules documented in the application reference. We compare the outputs of these filters with the corresponding outputs obtained using the OpenCV library. As the Wiener filter is not directly available in OpenCV, it was obtained in the standard local-statistics form, using two OpenCV box-filter operations to estimate the local mean and the local mean of squared intensities, followed by pointwise computation of the local variance, Wiener gain, and output.

Table 1 summarizes the comparison between images filtered by noncausal deterministic integer implementations of advanced filters and images filtered using the corresponding floating-point OpenCV implementations. Because the purpose of this comparison is not to match individual parameter settings exactly, but to verify whether whole filter families behave reasonably after integer reformulation, the table reports family-wise first, second, and third quartiles of PSNR values. The quartiles are computed over all configurations of a given filter family listed in Appendix A-B and over all GS2noise images.

The results in Table 1 indicate that the deterministic integer full-window implementations remain sufficiently close to their floating-point OpenCV counterparts for the purposes of the present study. The median PSNR is about 50 dB or higher for all four filter families, and it is substantially higher for guided and Wiener filtering. Thus, although the implementations are not numerically identical, the observed differences are small enough to treat the integer versions as adequate deterministic counterparts of the corresponding floating-point filters. The lowest median PSNR is obtained for bilateral filtering, not for NLM, which might be expected to be the most difficult case because of

TABLE 2. Causal deterministic integer filters, in `cropped-a` and `replicate-a` modes, compared with their noncausal full-window deterministic integer variants. Family-wise PSNR quartiles over all tested configurations on `GS2noise`.

Filter family	cropped-a			replicate-a		
	Q1 [dB]	median [dB]	Q3 [dB]	Q1 [dB]	median [dB]	Q3 [dB]
Median	26.96	28.73	31.50	27.73	30.00	31.94
Box	30.13	33.38	39.31	31.80	34.88	41.04
Weighted median, flat	27.36	29.32	33.44	28.36	30.68	34.31
Weighted median, power	30.61	40.36	∞	32.99	41.80	∞
Weighted median, exponential	28.21	30.56	35.90	29.33	31.86	36.75
Directional median, MAD	23.37	25.40	28.36	22.77	26.46	29.23
Directional median, SUMABS	24.76	27.20	30.18	24.45	27.27	29.94
Alpha-trimmed mean	28.42	30.10	33.50	29.13	31.79	34.24
Bilateral	34.55	39.86	46.38	34.56	39.43	45.73
Guided	32.54	37.54	45.36	34.76	39.71	46.87
Wiener	32.94	37.63	43.05	32.86	37.94	43.32
NLM	33.21	38.38	46.78	32.57	37.76	44.81

its complexity; the bilateral median is lower than the NLM median by 4.50 dB. This suggests that the discrepancy may be related more to differences in the detailed formulation or optimization of bilateral filtering used in OpenCV than to replacing floating-point computations by deterministic integer arithmetic alone. This issue is not pursued further here, as exact reproduction of OpenCV outputs is not required for the present RDLS evaluation.

Table 2 then compares the outputs of the causal deterministic integer implementations with those of the corresponding noncausal full-window deterministic integer implementations. We report results for both `cropped` and `replicate` causal modes; for simplicity, only the `filtering-a` variant is included. For a given filter family and causal mode, the table reports first, second, and third quartiles of PSNR values computed over all configurations listed for this family in Appendix A-B and over all `GS2noise` images.

As expected, causalization changes the filter outputs more substantially than the integer reformulation of advanced full-window filters. Nevertheless, the median PSNR remains at least 25.40 dB for all filter families in `cropped-a` mode and at least 26.46 dB in `replicate-a` mode. The infinite Q3 values for the weighted median with power weights indicate perfectly identical compared outputs; in these cases, some parameter settings reduce the filter to an identity operation equivalent to `NONE`. The `replicate-a` mode gives higher median PSNR for most filter families, whereas `cropped-a` is better for bilateral and NLM filtering. This supports retaining both causal-window modes in the candidate filter set.

B. ATTAINABLE CODING GAINS OF RDLS-MODIFIED PREDICTION

We first present the coding gains attainable using RDLS-modified prediction for the `FULL`, `FAST`, and `VFAST` filter spaces. These are the gains obtained with the ideal selector, which performs an exhaustive search for the denoising method according to the actual target metric, either H_0 or the compression rate c_r , and not according to the single scalar noise descriptor σ_{milli} . Table 3 reports the average gains for the `GS2noise` and `A3series` sets. The corresponding baseline values, i.e., the average H_0 and

TABLE 3. Average H_0 and c_r gains due to RDLS-modified prediction obtained with the ideal selector on the `GS2noise` and `A3series` sets.

Filter space	GS2noise		A3series	
	H_0 gain [%]	c_r gain [%]	H_0 gain [%]	c_r gain [%]
FULL	6.754	5.456	5.916	5.286
FAST	6.642	5.340	5.614	4.958
VFAST	6.025	5.029	4.741	4.638

Note: The average baseline rates for regular MED prediction were, respectively, $H_0 = 6.160$ bpp and $c_r = 6.230$ bpp for `GS2noise`, and $H_0 = 5.352$ bpp and $c_r = 5.377$ bpp for `A3series`.

c_r obtained for regular MED prediction without RDLS, or equivalently with the `NONE` filter, are given in the table note. These ideal-selector gains are subsequently used to assess the effectiveness of practical filter-selection models. At the same time, they show the average improvement that could be obtained by per-image exhaustive filter search.

The gains in Table 3 show that even the smallest candidate space retains a large part of the attainable improvement. The `VFAST` space, which contains only `NULL`, `NONE`, and box-filter configurations, reaches 89.2% and 92.2% of the `FULL` gain on `GS2noise` for H_0 and c_r , respectively, and 80.1% and 87.7% on `A3series`. This is better than expected for such a restricted space and suggests a possible practical alternative: instead of selecting the filter only from the global σ_{milli} estimate, the encoder could perform exhaustive search over a very small, possibly further ablated, `VFAST` space and signal the selected configuration to the decoder. This result suggests that simple filters may be sufficient for a practically useful variant of RDLS-modified prediction. By contrast, the advantage of `FULL`, which includes the NLM filter, over `FAST` is small in relation to the additional complexity: it is only 0.11–0.12 percentage points on `GS2noise` and 0.30–0.33 percentage points on `A3series`. Therefore, as anticipated in Section III-C, `FULL` should be treated mainly as a reference for simpler spaces rather than as a practical option for RDLS-modified prediction.

C. EFFECTIVENESS OF MODELS TRAINED ON `GS2NOISE`

For the presentation of detailed results of adaptive filter selection, we choose one fixed representative of each model

TABLE 4. Average gains of GS2noise-trained models for both training objectives and both evaluation sets. Here, F is the ratio of the reported gain to the corresponding ideal-selector gain.

Model family	Space	Models trained using H_0								Models trained using c_r							
		GS2noise LOGO-CV				A3series				GS2noise LOGO-CV				A3series			
		H_0 gain [%]	F	c_r gain [%]	F	H_0 gain [%]	F	c_r gain [%]	F	H_0 gain [%]	F	c_r gain [%]	F	H_0 gain [%]	F	c_r gain [%]	F
Lookup	FULL	4.280	0.63	3.453	0.63	4.810	0.81	3.922	0.74	2.023	0.30	3.355	0.61	3.532	0.60	4.449	0.84
Piecewise	FULL	4.567	0.68	3.638	0.67	5.183	0.88	4.269	0.81	3.305	0.49	3.766	0.69	4.179	0.71	4.571	0.86
Regression	FULL	4.571	0.68	3.599	0.66	5.042	0.85	4.056	0.77	3.917	0.58	4.101	0.75	4.175	0.71	4.401	0.83
Lookup	FAST	3.934	0.59	3.028	0.57	4.540	0.81	3.478	0.70	1.143	0.17	3.043	0.57	1.704	0.30	3.820	0.77
Piecewise	FAST	4.151	0.63	3.122	0.58	4.755	0.85	3.577	0.72	2.854	0.43	3.549	0.66	3.167	0.56	4.004	0.81
Regression	FAST	4.290	0.65	3.180	0.60	4.719	0.84	3.539	0.71	3.309	0.50	3.808	0.71	3.508	0.62	3.880	0.78
Lookup	VFAST	2.685	0.45	2.653	0.53	2.894	0.61	2.818	0.61	0.263	0.04	2.773	0.55	-2.627	-0.55	2.252	0.49
Piecewise	VFAST	3.303	0.55	2.831	0.56	3.177	0.67	2.942	0.63	1.453	0.24	3.130	0.62	1.119	0.24	3.548	0.76
Regression	VFAST	3.460	0.57	2.851	0.57	3.435	0.72	2.726	0.59	2.026	0.34	3.449	0.69	1.814	0.38	3.414	0.74
Average		3.916	0.60	3.150	0.60	4.284	0.78	3.481	0.70	2.255	0.34	3.442	0.65	2.286	0.40	3.815	0.77

family introduced in Section III-D: the lookup model, the four-segment piecewise model, and the logarithmic quadratic regression model. This choice is not intended to identify a separate best variant for each table entry. The final screening of all tested variants showed that several piecewise and regression variants perform similarly; therefore, reporting one representative from each family gives a more compact comparison. The four-segment piecewise model is used as an intermediate-complexity representative of the piecewise family. The logarithmic quadratic regression model `quad_log` is retained as a stable representative of the regression family: although it is not uniformly the best in every configuration, it remains among the best-performing variants, especially for models trained using the actual compression rate c_r , and is used consistently in Appendix B and in the subsequent analysis.

Table 4 reports the average gains obtained by models trained on the GS2noise set using either H_0 or c_r as the optimization objective. They are evaluated using both metrics on both image sets: GS2noise, using LOGO-CV testing, and A3series, using the model trained on all GS2noise images. For each gain, we also report the fraction F of the ideal-selector gain, defined as the ratio of this gain to the gain of the corresponding ideal selector from Table 3, for the same image set, filter space, and metric. Averages over the selected model families and filter spaces are included to simplify the analysis of model generalization and cross-dataset transfer.

The results in Table 4 indicate that all three reported model families are useful, although the differences among the best variants are moderate. The lookup model is a surprisingly competitive baseline, confirming that the global noise descriptor σ_{milli} captures a large part of the information needed for filter selection. The piecewise and regression models remain useful, but their additional gain over lookup is not uniform; their main advantage is better stability in cross-dataset and cross-metric tests rather than a large systematic improvement in every case.

The models trained using H_0 generalize reasonably well to the actual compression rate and transfer well to A3series. On average, the selected models recover $F = 0.60$ of the ideal-selector gain on GS2noise for both H_0 and c_r , and

$F = 0.78$ and $F = 0.70$ on A3series for H_0 and c_r , respectively. The weakest reported H_0 -trained case is the VFAST lookup model on GS2noise, for which $F = 0.45$ for H_0 . However, for the piecewise and regression models, and especially after transfer to A3series, the recovered fraction of the ideal-selector gain is higher. This behavior supports the use of σ_{milli} as a practical control variable, while also showing that a single global descriptor does not capture all image-dependent information used by the ideal selector.

The models trained directly using c_r are more appropriate for the practical compression-rate objective. Their average c_r gains are larger than for the H_0 -trained models: 3.442% on GS2noise and 3.815% on A3series, corresponding to $F = 0.65$ and $F = 0.77$, respectively. However, the same models generalize much worse to H_0 , with average fractions $F = 0.34$ and $F = 0.40$ for GS2noise and A3series, respectively. The most pronounced example is the VFAST lookup model, whose H_0 gain on A3series is negative. This shows that the memoryless entropy of the residual image and the actual EBCOT-coded rate are related, but not interchangeable, optimization objectives.

The selected `quad_log` models are presented in Appendix B and are used below as representative regression models. The corresponding model tables map possible σ_{milli} values to complete filter configurations; decoder-side operation therefore does not require independent floating-point regression evaluation. These tables also show that the models switch to the NULL case above sufficiently high σ_{milli} thresholds, especially for models optimized for c_r . Since NULL suppresses MED prediction instead of merely denoising its arguments, this behavior motivates the more detailed filter-selection and NULL-frequency analysis in Section V-E.

We also compared `filtering-abc` with the simpler `filtering-a`. To estimate its effect, we additionally evaluated simplified filter spaces in which only `filtering-a` was allowed. For GS2noise, this restriction reduced the attainable ideal-selector gain by about 0.13–0.19 percentage points, depending on the filter space and the evaluated criterion. For A3series, the mean decrease was slightly

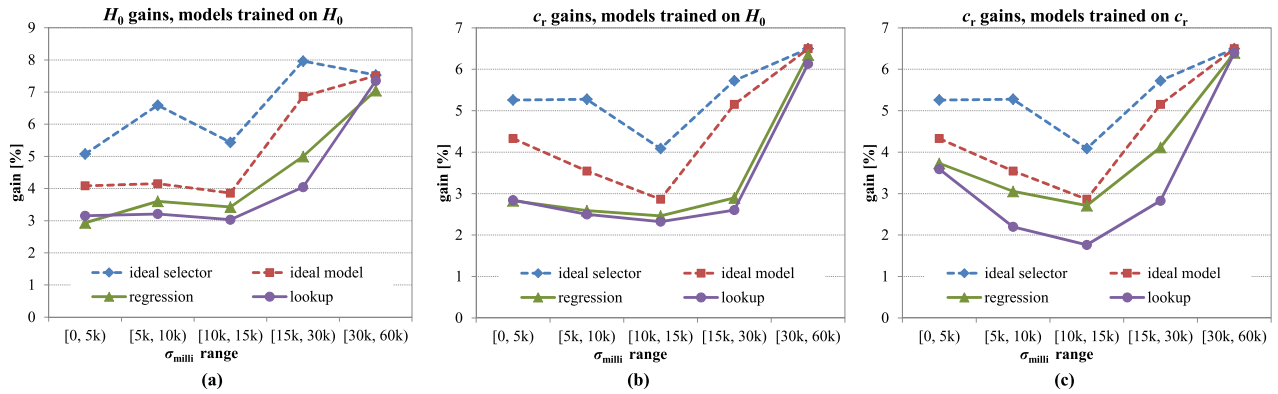


FIGURE 5. Coding gains for the GS2noise set as a function of σ_{milli} for the FAST filter space: (a) H_0 gains for models trained on H_0 , (b) c_r gains for models trained on H_0 , and (c) c_r gains for models trained on c_r . The range counts, from left to right, are 13, 26, 16, 23, and 12.

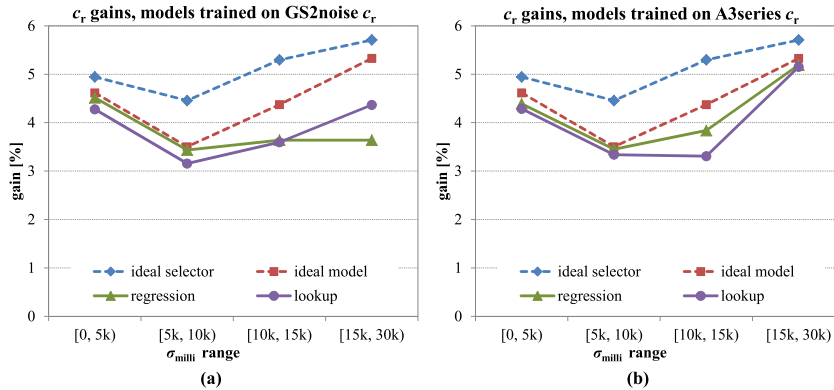


FIGURE 6. Compression-rate gains for the A3series set as a function of σ_{milli} for the FAST filter space, obtained using models trained on c_r of (a) GS2noise images and (b) A3series images. The range counts, from left to right, are 51, 49, 19, and 25.

larger, about 0.19 percentage points, with a wider spread. Thus, `filtering-abc` is useful, but its effect is moderate and comparable to the small FULL-FAST gap.

D. GAIN DEPENDENCE ON σ_{MILLI}

Figures 5 and 6 show the dependence of coding gain on σ_{milli} for the FAST filter space. The plots compare two practical models, namely the lookup model and the logarithmic regression model, with two reference curves: the ideal selector, which chooses the best filter independently for each image, and the additional ideal model, which is restricted to σ_{milli} and chooses the best average filter for each exact value of σ_{milli} . To improve readability, the results are averaged over predefined σ_{milli} intervals. Fig. 5 shows behavior on the main model-development dataset, GS2noise, whereas Fig. 6 shows gains obtained on A3series by models trained on GS2noise, as well as those obtained by models trained and tested on A3series using LOGO-CV. This distinction separates cross-dataset transfer effects from the intrinsic behavior of the model family.

Several observations follow from these figures. First, the gain does not increase monotonically with σ_{milli} in all cases. For GS2noise, the clearest growth is observed for the H_0 gain of the H_0 -trained models, whereas the c_r gains are more nearly flat in the lower and middle ranges and become

largest mainly in the highest range. For A3series, the highest range also gives larger gains when the models are trained on A3series, but the model trained on GS2noise transfers less uniformly. In particular, its regression curve for c_r does not improve in the highest A3series range. Thus, RDLS-modified prediction remains useful over the whole plotted range, including lower σ_{milli} values, but the high-noise advantage is concentrated mainly in the highest σ_{milli} interval rather than increasing steadily with σ_{milli} .

The ideal model provides an additional diagnostic reference: unlike the ideal selector, it selects only the best average filter for a given value of σ_{milli} , not the best filter separately for each image. For A3series, the relation between the practical models and this reference is consistent with that observed for GS2noise. The practical curves remain below the ideal model, but they follow it reasonably closely, especially in the highest GS2noise range and when both training and testing are performed on A3series. The remaining gap to the ideal selector therefore indicates that part of the image-dependent information useful for filter selection is not captured by σ_{milli} alone.

The gap between the ideal model and the ideal selector is important because the ideal model is already an oracle with respect to the scalar descriptor: it uses the data to choose the best average configuration for each value of

TABLE 5. Average compression-rate gains for low-noise images, obtained with models trained using c_r on the GS2noise set and the FAST filter space.

Model	GS2noise without AWGN		ISO 100 subset of A3series	
	c_r gain [%]	F	c_r gain [%]	F
Lookup	1.905	0.45	3.588	0.76
Regression	2.158	0.51	4.248	0.90

σ_{milli} . Therefore, the remaining gap cannot be removed only by changing the functional form of the practical lookup, piecewise, or regression model. It indicates that images or components with similar global estimated noise levels may still require different filters due to factors such as texture, edge structure, histogram properties, or color component.

To examine the behavior of the practical models in the low-noise regime, we additionally evaluated the same two FAST-space models as above, namely the lookup and regression models trained on GS2noise using c_r , on two restricted image subsets: the original GS2noise images without added synthetic noise and the ISO 100 subset of A3series. For GS2noise without AWGN, the average c_r without RDLS was 4.710 bpp and the ideal-selector gain was 4.217%. For the ISO 100 subset of A3series, the corresponding values were 4.090 bpp and 4.696%, respectively. Table 5 reports the corresponding gains and fractions F of the ideal-selector gains.

The results show that RDLS-modified prediction and the proposed practical models also yield noticeable gains in the low-noise regime. For GS2noise without added AWGN, the lookup and regression models recover a moderate fraction of the ideal-selector gain, with $F \geq 0.45$ and a slight advantage of the regression model. For the ISO 100 subset of A3series, both models recover a larger fraction, $F \geq 0.76$, and the logarithmic regression model performs better. Thus, the method appears useful also at low noise levels, not only in the higher-noise ranges highlighted above.

E. FILTER-SELECTION DIAGNOSTICS AND NULL BEHAVIOR

Figure 7 shows what the ideal selector chooses in the FAST space when the objective is the actual compression rate c_r . The strongest tendency is clear in both datasets: bilateral filtering dominates in the low and middle σ_{milli} ranges, whereas the NULL case becomes dominant at the highest estimated noise levels represented in each dataset. However, the relation is not deterministic. In GS2noise, NULL is already selected for some images in the lower ranges, while in the intermediate [15k, 30k] range the ideal selections are split among bilateral, guided, and NULL. This illustrates both the usefulness of σ_{milli} as a control variable and its limitations in reproducing ideal per-image decisions.

The NULL case is therefore not a marginal special case. For the FAST space and the c_r objective, it is selected by the ideal per-image selector for 27.8% of the GS2noise images and for 19.4% of the A3series components. Its frequency rises to 91.7% in the highest GS2noise range and

to 76.0% in the highest A3series range, corresponding to 11 of 12 images and 19 of 25 components, respectively. Thus, these results support the interpretation of NULL as an adaptive suppression of ineffective MED prediction when denoised predictor arguments no longer improve the coded residual rate. For A3series, the high-ISO NULL selections were also component-dependent: they occurred mainly in the R and B components and only rarely in G, consistently with the dataset construction described in Section IV-C, where G is obtained by averaging the two green subpixels of each RGB Bayer block.

Table 6 summarizes the same behavior from the point of view of a compact descriptor/model gap analysis. In GS2noise with the FAST space, the ideal per-image gain is 5.340%, whereas the ideal model reaches 4.341% ($F = 0.81$). The practical logarithmic regression model reaches 3.808% ($F = 0.71$), with an average regret of 0.089 bpp relative to the ideal selector. Thus, a substantial part of the gap is already present when the selector is constrained to use only one global descriptor, before considering the particular functional form of the regression model.

Two recurring failure modes help explain the observed regret. First, the model may select a smoothing filter, most often bilateral, although the ideal selector chooses NULL; this occurs when the global noise estimate is not high enough to trigger the learned NULL transition, but the actual residual coding behavior of the image favors suppressing MED prediction. Second, the opposite error also occurs for texture-rich or strongly structured images, where the model switches to NULL although a denoising filter is still beneficial. In cross-dataset testing on A3series, the GS2-trained FAST regression model underselects NULL: the ideal selector chooses NULL for 19.4% of the A3series components, whereas the transferred regression model chooses it for none of them. This is consistent with the interpretation that the transfer gap is not attributable solely to estimator error and suggests that richer descriptors or local/block-adaptive selection may be more effective than further refining a single global σ_{milli} regression.

Overall, σ_{milli} captures the broad transition from smoothing filters to NULL, but it does not distinguish all image-dependent cases within the same estimated-noise range. This motivates the preliminary block-adaptive VFAST experiment in the next subsection and, more generally, future selectors using additional low-cost global or local descriptors.

F. PRELIMINARY BLOCK-ADAPTIVE VFAST SELECTION

Table 7 summarizes the preliminary exhaustive block-adaptive VFAST experiment. The best average c_r result was obtained for 64×64 blocks on both datasets as an exploratory post-hoc best among the seven tested block sizes. With side information included, block-adaptive VFAST achieved a c_r gain of 4.942% on GS2noise and 4.460% on A3series. This substantially improves over the corresponding global VFAST regression models, which gave 3.449% and 3.623%, respectively. The adaptive results also approach the global

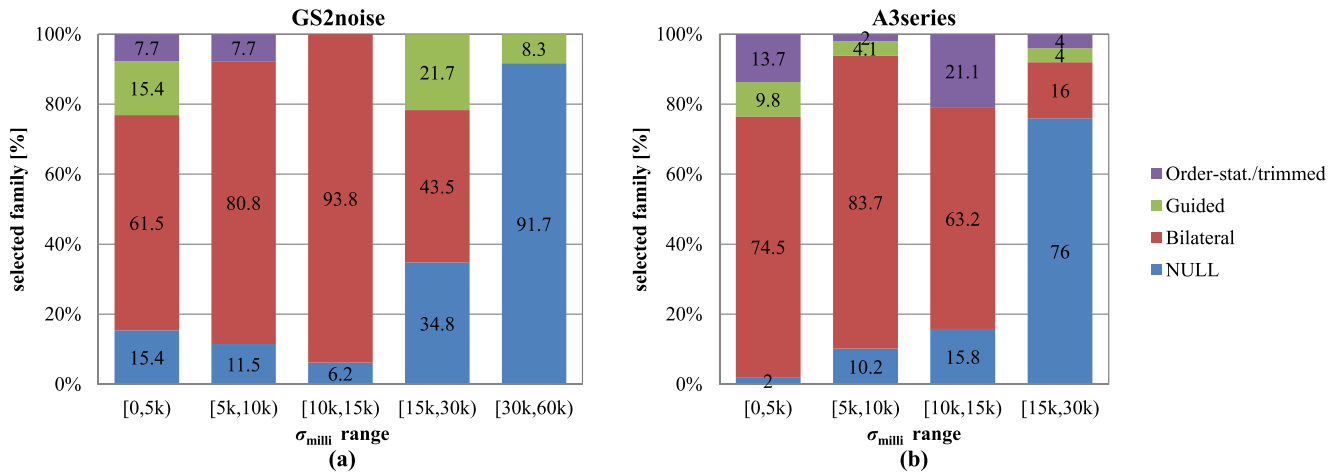


FIGURE 7. Distribution of filter families selected by the ideal per-image selector as a function of the estimated noise-level range, for the FAST filter space and the compression-rate objective c_r : (a) GS2noise and (b) A3series. Percentages are computed within each σ_{milli} range. The range counts, from left to right, are 13, 26, 16, 23, and 12 for GS2noise, and 51, 49, 19, and 25 for A3series. Minor order-statistics and trimmed-mean families are grouped for readability.

TABLE 6. Compact gap analysis for the compression-rate objective c_r . The ideal model is the σ_{milli} -restricted reference defined in Section V-D. F is the fraction of the corresponding ideal-selector gain, and regret is the average excess rate of the regression model over the ideal selector. For the A3series GS2noise-trained row, the ideal-selector and ideal-model references are computed on the A3series evaluation set, whereas only the regression model is transferred from GS2noise.

Dataset / evaluation	Space	Ideal gain [%]	Ideal model gain [%] (F)	Regression gain [%] (F)	Regret [bpp]	NULL ideal/reg. [%]
GS2noise LOGO-CV	FAST	5.340	4.341 (0.81)	3.808 (0.71)	0.089	27.8 / 22.2
GS2noise LOGO-CV	VFAST	5.029	3.959 (0.79)	3.449 (0.69)	0.091	30.0 / 22.2
A3series LOGO-CV	FAST	4.958	4.328 (0.87)	4.137 (0.83)	0.045	19.4 / 18.8
A3series LOGO-CV	VFAST	4.638	3.911 (0.84)	3.623 (0.78)	0.054	22.2 / 18.8
A3series GS2noise-trained	FAST	4.958	4.328 (0.87)	3.880 (0.78)	0.064	19.4 / 0.0

TABLE 7. Preliminary exhaustive block-adaptive VFAST results. Gains include the fixed one-byte-per-block side information. The tested block sizes were 8, 16, 32, 64, 128, 256, and 512; the reported 64×64 block size is the exploratory post-hoc best-of-seven choice giving the best average c_r gain.

Dataset	Side [bpp]	H_0 gain [%]	c_r gain [%]	Global reg. c_r gain [%]	Global ideal c_r gain [%]
GS2noise	0.0020	4.990	4.942	3.449	5.029
A3series	0.0021	0.777	4.460	3.623	4.638

per-image VFAST ideal-selector references, 5.029% and 4.638%, although the two selectors are not directly equivalent because the block-adaptive method uses a larger local decision space.

The global H_0 trend was less aligned with c_r , especially on A3series, because the final residual image is assembled from blocks with different residual distributions. This motivates further adaptive-selection studies.

G. COLOR-COMPONENT VALIDATION WITH A MODULAR REVERSIBLE COLOR TRANSFORM

Table 8 shows that RDLS-modified prediction remains effective after the modular reversible color transform. For HP+mRCT, the GS2noise-trained FAST regression model gives a c_r gain of 4.252%, compared with 3.880% for the RGB-component reference. The recovered fraction of the FAST ideal-selector gain is also higher, $F = 0.937$ instead of $F = 0.783$. For mRCT, without HP before the color

TABLE 8. Color-component validation on A3series for the compression-rate objective c_r , using FAST-space models trained on GS2noise. F is the fraction of the FAST ideal-selector gain recovered by the regression model.

Representation	MED+ EBCOT c_r [bpp]	Ideal gain [%]	Lookup gain [%]	Reg. gain [%]	F	Reg. c_r [bpp]
HP-packed RGB	5.377	4.958	3.820	3.880	0.783	5.168
HP+mRCT	5.494	4.540	3.893	4.252	0.937	5.261
mRCT	5.817	4.332	3.823	3.971	0.917	5.586

transform, the regression gain remains positive at 3.971% ($F = 0.917$).

The modular transform itself is not a better baseline representation for the full A3series set: the MED+EBCOT rate increases from 5.377 bpp for HP-packed RGB components to 5.494 bpp for HP+mRCT and 5.817 bpp for mRCT, and the final RDLS rates remain worse than for RGB. Thus, this compatibility check is not a recommendation of the modular reversible color transform for A3series; rather, it shows that RDLS-PredMED can be applied without modification to components produced by a reversible color transform.

H. COMPARISON WITH STANDARD LOSSLESS IMAGE CODECS

In Table 9, we compare actual compression rates for GS2noise images, averaged over σ_{milli} ranges and over all images, obtained using RDLS-modified prediction coupled

TABLE 9. Comparison of average compression rates for $GS2noise$ images. Rates are reported for σ_{milli} ranges and for all images. Standard lossless codecs are compared with unmodified MED prediction and with RDLS-modified MED prediction using logarithmic regression models and the FAST and VFAST filter spaces.

σ_{milli} range	JPEG-LS	JPEG 2000	JPEG XL				unmodified	RDLS-MED	RDLS-MED
							MED+EBCOT	FAST+EBCOT	VFAST+EBCOT
	[bpp]	[bpp]	e1	e4	e7	e10	[bpp]	[bpp]	[bpp]
0–5k	4.190	4.247	4.633	4.145	4.064	3.992	4.425	4.261	4.299
5k–10k	5.289	5.370	5.685	5.287	5.118	4.959	5.522	5.353	5.384
10k–15k	6.020	6.106	6.378	6.041	5.905	5.812	6.273	6.103	6.120
15k–30k	6.811	6.878	7.093	6.788	6.619	6.464	7.061	6.768	6.773
30k–60k	7.776	7.852	8.070	7.801	7.652	7.288	8.069	7.549	7.549
All	5.981	6.055	6.334	5.975	5.827	5.666	6.230	5.983	6.002

with the EBCOT entropy coder in the same way as in the previous sections, with standard lossless image codecs. The external references include JPEG-LS, JPEG 2000, and JPEG XL. For JPEG XL, we report four effort settings, $e \in \{1, 4, 7, 10\}$, where $e = 1$ is the fastest setting and $e = 10$ is the most complex setting.

All codecs in Table 9 were evaluated on the same histogram-packed $GS2noise$ images as the RDLS experiments. The histogram-signaling cost was not added to the reported rates, consistently with Section IV-A. The codec implementations and command-line settings used to obtain the rates are described in Section IV-F. For RDLS-modified prediction, the table reports variants with denoising filters selected from the FAST and VFAST filter spaces by the logarithmic regression model trained on the compression rate of $GS2noise$ and evaluated using LOGO-CV. To distinguish the use of only the JPEG 2000 entropy coder EBCOT from full JPEG 2000 with the standard DWT stage, which is reported in the JPEG 2000 column, these variants are denoted RDLS-MED FAST+EBCOT and RDLS-MED VFAST+EBCOT. We also include rates obtained by coding the prediction residuals from unmodified MED prediction, denoted unmodified MED+EBCOT.

The simple MED+EBCOT scheme is not a competitive standalone alternative to standard image codecs. On average, it is worse than JPEG 2000 by 0.175 bpp and worse than JPEG-LS by 0.249 bpp, and it is also worse than JPEG XL at efforts e4, e7, and e10. It is better only than the fastest tested JPEG XL setting, e1. This was expected because EBCOT is designed as the core JPEG 2000 entropy-coding algorithm for DWT-transformed subbands [47], rather than being tailored to MED residuals. In this comparison, MED+EBCOT therefore serves mainly as a baseline for assessing the effect of applying RDLS to prediction.

Applying RDLS substantially shifts this baseline. The FAST and VFAST variants reduce the average rate of MED+EBCOT by 0.247 and 0.228 bpp, respectively. On average, both RDLS variants outperform JPEG 2000 and are close to JPEG-LS and JPEG XL e4: the FAST variant is only 0.002 bpp worse than JPEG-LS and 0.008 bpp worse than JPEG XL e4, while the corresponding differences for VFAST are 0.021 and 0.027 bpp. JPEG XL at efforts e7 and e10 still gives lower average rates.

The advantage of RDLS-modified prediction becomes clearer in the mid- and high-noise ranges. In the 15k–30k range, both RDLS variants outperform JPEG-LS, JPEG 2000, and JPEG XL e4, although they remain worse than JPEG XL e7 and e10. In the highest 30k–60k range, both variants improve over unmodified MED+EBCOT by 0.520 bpp and outperform JPEG-LS, JPEG 2000, JPEG XL e4, and JPEG XL e7; however, JPEG XL e10 is still better by 0.261 bpp. Thus, the largest practical effect of RDLS appears for images with the highest estimated noise level.

Overall, the comparison shows how far RDLS shifts a simple MED+EBCOT predictive baseline. The VFAST variant retains most of the FAST effect, being only 0.019 bpp worse on average and identical in the highest σ_{milli} range, while using a much simpler filter space.

I. RUNTIME MEASUREMENTS FOR THE VFAST PATH

To complement the coarse operation-count analysis from Section III-C and to address practical usefulness, we measured the single-threaded runtime of the main operations needed by the MED baseline and the VFAST RDLS path. The benchmark was run on all 144 A3series components, chosen because their much larger size gives more stable timing measurements than the smaller $GS2noise$ images, using a Visual Studio 2022 Release x64 build on a Ryzen 7 5700G processor with turbo disabled at 3.8 GHz. For each component, one warm-up repetition was discarded and ten repetitions were measured. Internal operations were measured as memory-to-memory kernels writing to separate output buffers, whereas Kakadu and OpenJPEG were measured as file-to-file codec calls. All checksum-consistency checks passed, and decoded outputs of the external codecs were verified bit-exactly. Table 10 reports the results; it does not include the encoder-side exhaustive search used by the block-adaptive experiment in Section V-F; the H_0 and standard JPEG 2000 timings are included only as auxiliary diagnostics. The filtering-abc timings are conservative because the three argument variants are computed independently; an integrated codec could share part of this work.

The σ_{milli} estimator is inexpensive, requiring 1.45 ms/Mpix, less than one third of the time required for MED residual generation. Box+MED required 19.14–24.44 ms/Mpix for

TABLE 10. Single-threaded runtime measurements for external codecs and selected MED/VFAST operations on the A3series components. Times are reported as means in ms/Mpix. In the Type column, *file* denotes file-to-file codec calls, whereas *memory* denotes memory-to-memory kernels. External-codec and predictive-operation timings are for histogram-packed components; HP time was measured separately. For Box+MED, each interval spans the mean times obtained for the tested box radii $r \in \{1, 2, 3, 4, 6, 8, 12\}$ with center weight $c = 4$.

Type	Operation	Encode/ forward [ms/Mpix]	Decode/ inverse [ms/Mpix]
file	Kakadu EBCOT, Clevels = 0	72.03	79.58
file	Kakadu JPEG 2000	81.74	89.72
file	OpenJPEG JPEG 2000	121.41	137.18
memory	HP	0.99	0.72
memory	σ_{milli} estimator	1.45	–
memory	H_0 , image / MED residual	0.35 / 0.47	–
memory	MED baseline	4.92	6.95
memory	NULL path	0.05	0.54
memory	Box+MED, filtering-a	19.14–24.44	19.53–24.92
memory	Box+MED, filtering-abc	37.89–53.51	37.82–53.55

filtering-a and 37.89–53.51 ms/Mpix for filtering-abc in the forward path, with similar inverse times. Thus, introducing the VFAST box filtering path increases the MED-prediction kernel time by a factor of a few, rather than by an order of magnitude. As contextual non-end-to-end references, these RDLS-related memory-to-memory kernel times remain numerically below the times measured for the file-to-file Kakadu EBCOT calls, suggesting that the additional kernel cost of the practical RDLS path is moderate in this benchmark.

VI. CONCLUSION

Building on the lifting interpretation of prediction-error computation, we introduced an RDLS-modified MED predictor that replaces the predictor arguments with causally and deterministically denoised variants. The resulting RDLS-modified MED predictor was evaluated on scalar 8-bit photographic image components with synthetic and real acquisition noise, using residual memoryless entropy and the bit rate obtained with an EBCOT coder. In the main model-based setting, the denoising filter was selected for each image/component using the estimated global noise-level descriptor σ_{milli} .

The results show that RDLS-modified MED prediction can improve the evaluated MED+EBCOT predictive-coding pipeline, especially for images with nonnegligible noise. In LOGO-CV on the GS2noise dataset, the practical FAST and VFAST variants improve the average bit rate by 0.247 and 0.228 bpp, respectively, over unmodified MED+EBCOT; in the highest estimated-noise range, both improve it by 0.520 bpp. The gains are still visible in the low-noise regime: for original GS2noise images without added AWGN, the lookup and regression models give c_r gains of 1.905% and 2.158%, respectively, whereas for the ISO 100 subset of A3series they give 3.588% and 4.248%, respectively. Thus, the method remains useful at low noise levels, but its effect is largest at high estimated noise levels.

Adaptive models driven only by σ_{milli} recover substantial fractions of the ideal-selector gain. For models trained directly on c_r , the average recovered fractions are $F = 0.65$ on GS2noise and $F = 0.77$ on A3series; models trained on H_0 transfer reasonably to c_r but are less suitable for this practical objective. The remaining gaps, both to the ideal selector and to the ideal model driven only by σ_{milli} , indicate that this single global descriptor is a main limitation of the current selection path. At the same time, VFAST retains much of the FAST effect. The preliminary exhaustive block-adaptive VFAST experiment further shows that local decisions can improve over global regression and approach the global VFAST ideal-selector reference in terms of c_r , although the decision spaces are not equivalent. The A3series reversible color transform validation confirms that RDLS-PredMED remains effective on transformed color components.

Against standard codecs, RDLS substantially improves the simple MED+EBCOT baseline. On average, the FAST variant is close to JPEG-LS and JPEG XL effort 4, whereas higher-effort JPEG XL remains better; in the highest σ_{milli} range, the FAST and VFAST variants outperform JPEG-LS, JPEG 2000, JPEG XL effort 4, and JPEG XL effort 7. Thus, the comparison should be read as evidence that RDLS can strongly improve a simple predictive baseline, not as a claim of a new standalone state-of-the-art codec.

The results motivate further work on improved global or local filter-selection methods, extensions to predictor selection, ablation of filter parameter sets, families, and modes, and production-level integration of the benchmarked VFAST kernels. A separate direction is the design of causal, deterministic, fixed-point adaptations of more recent denoising architectures for use as RDLS filters.

APPENDIX A

FILTERS USED IN THIS STUDY

A. SHORT CHARACTERISTICS OF THE FILTER FAMILIES

Below, we briefly characterize the filter families used in the study, excluding the special NULL and NONE cases already described in Section II-B.

The box filter is a very simple local averaging filter. It averages the samples contained in the filter window, optionally assigning increased weight to the central sample. It reduces random fluctuations by simple smoothing, which is useful mainly for approximately Gaussian noise. At the same time, because it does not distinguish between smooth regions and edges, it may blur local image structure and therefore may be less suitable near sharp transitions.

The median filter is a simple order-statistics filter that replaces the central sample by the median of the samples in the window. Besides simplicity, its main practical advantage is robustness to outliers and impulse-noise disturbances, while it also tends to preserve edges better than ordinary averaging. In this study, it is useful as a simple nonlinear

TABLE 11. Filter parameters used in the study.

Filter family / variant	Parameters	Parameter values used
NULL	–	–
NONE	–	–
Median	Radius r	$r \in \{1, 2, 3, 4, 6, 8, 12\}$
Box	Radius r , center weight c	$r \in \{1, 2, 3, 4, 6, 8, 12\}$, $c \in \{1, 2, 4, 8, 16, 32, 64, 128, 256\}$
Weighted median / flat	Radius r , center weight c	$r \in \{1, 2, 3, 4, 6, 8, 12\}$, $c \in \{1, 2, 4, 8, 16\}$
Weighted median / power-distance	Radius r , center weight c , exponent p	$r \in \{1, 2, 3, 4, 6, 8, 12\}$, $c = 4$, $p \in \{1, 2, 3\}$
Weighted median / exponential-distance	Radius r , center weight c , decay parameter λ	$r \in \{1, 2, 3, 4, 6, 8, 12\}$, $c = 4$, $\lambda \in \{r/2, r, 2r\}$
Directional median / MAD scoring	Radius r , band thickness t	$r \in \{2, 3, 4, 6, 8, 12\}$, $t \in \{0, 1\}$
Directional median / SUM-ABS scoring	Radius r , band thickness t	$r \in \{2, 3, 4, 6, 8, 12\}$, $t \in \{0, 1\}$
Alpha-trimmed mean	Radius r , trimming fraction α	$r \in \{1, 2, 3, 4, 6, 8, 12\}$, $\alpha \in \{0, 0.02, 0.05, 0.10, 0.15, 0.20, 0.30, 0.40\}$
Bilateral	Radius r , color coefficient k_{color} , spatial scale σ_{space}	$r \in \{2, 3, 4\}$, $k_{\text{color}} \in \{0.3, 0.5, 0.75, 1.0, 1.25, 1.5, 2.0, 3.0, 6.0, 9.0, 12.0\}$; for $r = 2$: $\sigma_{\text{space}} \in \{1.0, 1.2, 1.8, 2.5, 3.5\}$; for $r = 3$: $\sigma_{\text{space}} \in \{1.2, 1.5, 2.5, 3.5, 5.0\}$; for $r = 4$: $\sigma_{\text{space}} \in \{1.5, 2.0, 3.0, 4.5, 6.5\}$
Guided	Radius r , coefficient k_{ϵ}	$r \in \{2, 3, 4, 6, 8, 12\}$, $k_{\epsilon} \in \{0.5, 0.75, 1.0, 1.5, 2.0, 3.0, 4.0, 6.0, 9.0\}$
Wiener	Radius r , coefficient k_{σ}	$r \in \{1, 2, 3, 4\}$, $k_{\sigma} \in \{0.5, 0.65, 0.8, 1.0, 1.25, 1.5, 2.0, 3.0\}$
NLM	Patch radius r , coefficient k_h , search-window side length s_w	$r \in \{2, 3, 4\}$, $k_h \in \{0.3, 0.5, 0.65, 0.8, 1.0, 1.25, 1.5, 2.0, 2.5\}$, $s_w \in \{11, 21, 31\}$

For window radius r , the window side length is $2r + 1$ and the window size is $(2r + 1)^2$. For the bilateral filter, the actual call uses $\sigma_{\text{color}} = k_{\text{color}}\sigma$, while σ_{space} is used directly. For the guided filter, the actual regularization parameter is $\epsilon = (k_{\epsilon}\sigma)^2$. For the Wiener filter, the noise variance is $(k_{\sigma}\sigma)^2$. For NLM, the actual call uses $h = k_h\sigma$, and k_p and k_s used in Section III-C are $(2r + 1)^2$ and s_w^2 , respectively.

TABLE 12. Selected logarithmic quadratic regression models trained on GS2noise for the FULL filter space.

Optimized for H_0			Optimized for c_r		
σ_{milli} range	Mode	Filter and parameters	σ_{milli} range	Mode	Filter and parameters
0–1572	replicate-abc	NLM; $r = 4$, $k_h = 0.8$, $s_w = 11$	0–3669	replicate-abc	NLM; $r = 3$, $k_h = 2$, $s_w = 21$
1573–2621	replicate-abc	NLM; $r = 2$, $k_h = 1$, $s_w = 11$	3670–17298	cropped-abc	NLM; $r = 2$, $k_h = 1.5$, $s_w = 21$
2622–4717	replicate-abc	NLM; $r = 2$, $k_h = 1$, $s_w = 21$	17299–23588	cropped-abc	NLM; $r = 3$, $k_h = 1.5$, $s_w = 21$
4718–5766	replicate-abc	NLM; $r = 2$, $k_h = 1.25$, $s_w = 11$	23589–267331	–	NULL
5767–12056	cropped-abc	NLM; $r = 2$, $k_h = 1.25$, $s_w = 11$			
12057–21491	cropped-abc	NLM; $r = 2$, $k_h = 1.25$, $s_w = 21$			
21492–30926	cropped-abc	NLM; $r = 2$, $k_h = 1.5$, $s_w = 11$			
30927–31975	replicate-abc	NLM; $r = 2$, $k_h = 1.5$, $s_w = 21$			
31976–33023	replicate-abc	Box; $r = 4$, $c = 4$			
33024–34071	replicate-a	Box; $r = 6$, $c = 16$			
34072–35120	replicate-abc	Box; $r = 6$, $c = 16$			
35121–267331	–	NULL			

TABLE 13. Selected logarithmic quadratic regression models trained on GS2noise for the FAST filter space.

Optimized for H_0			Optimized for c_r		
σ_{milli} range	Mode	Filter and parameters	σ_{milli} range	Mode	Filter and parameters
0–1572	replicate-abc	Guided; $r = 3$, $k_{\epsilon} = 1$	0–1572	replicate-abc	Guided; $r = 3$, $k_{\epsilon} = 3$
1573–4717	replicate-abc	Bilateral; $r = 4$, $k_{\text{color}} = 2$, $\sigma_{\text{space}} = 2$	1573–11008	replicate-abc	Bilateral; $r = 4$, $k_{\text{color}} = 6$, $\sigma_{\text{space}} = 2$
4718–5766	replicate-abc	Wiener; $r = 2$, $k_{\sigma} = 1.25$	11009–14153	cropped-abc	Bilateral; $r = 4$, $k_{\text{color}} = 6$, $\sigma_{\text{space}} = 1.5$
5767–12056	replicate-abc	Wiener; $r = 3$, $k_{\sigma} = 1.25$	14154–22539	cropped-abc	Bilateral; $r = 4$, $k_{\text{color}} = 6$, $\sigma_{\text{space}} = 2$
12057–16249	replicate-abc	Wiener; $r = 3$, $k_{\sigma} = 1.5$	22540–267331	–	NULL
16250–29878	replicate-abc	Wiener; $r = 4$, $k_{\sigma} = 1.5$			
29879–31975	replicate-abc	Box; $r = 4$, $c = 8$			
31976–33023	replicate-abc	Box; $r = 4$, $c = 4$			
33024–34071	replicate-a	Box; $r = 6$, $c = 16$			
34072–35120	replicate-abc	Box; $r = 6$, $c = 16$			
35121–267331	–	NULL			

baseline that may suppress local noise without introducing as much edge blurring as linear smoothing [39].

The alpha-trimmed mean filter combines averaging with trimming of extreme values. In this filter, the samples within

the window are sorted, a selected fraction of the smallest and largest values is discarded, and the remaining values are averaged. Consequently, the filter forms an intermediate case between the box filter and the median filter: it remains

TABLE 14. Selected logarithmic quadratic regression models trained on GS2noise for the VFAST filter space.

Optimized for H_0			Optimized for c_r		
σ_{milli} range	Mode	Filter and parameters	σ_{milli} range	Mode	Filter and parameters
0–2621	replicate-abc	Box; $r = 1, c = 16$	0–1572	replicate-abc	Box; $r = 3, c = 8$
2622–4717	replicate-abc	Box; $r = 1, c = 8$	1573–4717	replicate-abc	Box; $r = 2, c = 4$
4718–9959	replicate-abc	Box; $r = 1, c = 4$	4718–5766	replicate-abc	Box; $r = 2, c = 2$
9960–15201	replicate-abc	Box; $r = 1, c = 2$	5767–13104	replicate-abc	Box; $r = 2, c = 1$
15202–17298	cropped-abc	Box; $r = 1, c = 2$	13105–17298	replicate-abc	Box; $r = 3, c = 1$
17299–22539	replicate-abc	Box; $r = 2, c = 4$	17299–21491	cropped-abc	Box; $r = 3, c = 1$
22540–24636	replicate-abc	Box; $r = 2, c = 2$	21492–22539	cropped-abc	Box; $r = 4, c = 1$
24637–28830	replicate-abc	Box; $r = 3, c = 4$	22540–267331	–	NULL
28831–31975	replicate-abc	Box; $r = 4, c = 8$			
31976–33023	replicate-abc	Box; $r = 4, c = 4$			
33024–34071	replicate-a	Box; $r = 6, c = 16$			
34072–35120	replicate-abc	Box; $r = 6, c = 16$			
35121–267331	–	NULL			

effective for approximately Gaussian noise, while being less sensitive to outliers than simple averaging [39]. The box filter may be considered a special case of the alpha-trimmed mean filter.

The weighted median filter families extend the median idea by assigning different weights to samples before computing the order-statistics-based output; the median filter is a special case of these filters. In the flat variant, the central sample may simply receive a larger weight than the other samples. In the position-dependent variants, the weights decrease with distance according to a selected rule, namely a power or exponential function of the Chebyshev distance from the window center in our case. These filters retain the outlier robustness of median-like processing while allowing the center and nearby samples to influence the result to a greater extent, which makes them attractive when one seeks a compromise between noise reduction and preservation of local image structure [40], [41].

The directional median filter family exploits directional organization of the local neighborhood. Instead of treating all samples in the filter window uniformly, it evaluates several directional subsets, horizontal and vertical in our case with window corners ignored, and prefers subsets that are more consistent according to a given criterion. Such filters are intended to preserve edge-like structures better than isotropic order-statistics filters, since they can adapt their behavior to locally dominant directions [42].

The bilateral filter is a more advanced edge-preserving nonlinear smoother that combines spatial proximity with photometric similarity. In practice, nearby samples that are also similar in intensity have the greatest influence on the result, whereas samples across sharp intensity changes are suppressed. This makes the bilateral filter attractive for noisy images in which one wishes to reduce noise while avoiding excessive smoothing across edges [43].

The guided filter is another advanced edge-preserving smoothing method, based on a local model relating the output to a guidance image. In the present study, the image being filtered also serves as the guidance signal. The filter is well known for combining effective smoothing in relatively

uniform regions with good preservation of pronounced structural transitions, which makes it a natural candidate for noise-aware predictive coding [44].

The local adaptive Wiener filter employs local statistics, in particular the local mean and variance, to adapt the strength of smoothing to the observed signal variability. In relatively uniform regions, it tends to smooth more, while in regions of more pronounced local structure it tends to preserve more detail. This adaptive behavior makes it well suited to situations in which noise suppression and local-structure preservation must be balanced [45].

Finally, the nonlocal means (NLM) filter family is based on the observation that similar local patterns may occur at different positions within the image. Instead of relying only on a small local window, it usually compares rectangular patches of pixels within a larger search region, corresponding to the traditional filter window, and averages samples with weights reflecting patch similarity. This allows it to preserve repeated or structured image content particularly well, and it is therefore often very effective for Gaussian-like noise. It is the most computationally demanding filter family considered here. In the present study, it serves as the reference family used only within the FULL filter space [46].

B. FILTER PARAMETERS USED IN THE STUDY

This appendix lists the parameter ranges used for all filter families considered in the experiments. The ranges were chosen broadly enough to cover values regarded as practically useful, intentionally with some margin, since parameter values suitable for RDLS-modified prediction may differ from those typically preferred in conventional denoising applications. Because the adaptive models in Section III-D do not estimate filter parameters directly from σ_{milli} , but instead select from a finite set of filter configurations present in the training data, the parameter grids were also chosen to be relatively dense. For selected filters, the ranges were moderately extended after preliminary experiments. Table 11 summarizes the tested filter variants and their parameter values. The tested configurations correspond to the

deterministic integer or fixed-point implementations used in the experiments.

APPENDIX B

SELECTED LOGARITHMIC REGRESSION MODELS

Tables 12–14 report the selected logarithmic quadratic regression models trained on the entire GS_{2noise} set. The models were built separately for the FULL, FAST, and VFAST filter spaces and for the two optimization criteria, H_0 and c_r . Each table gives the filter configuration selected by the corresponding model as a function of σ_{milli} .

The supplementary CSV files specify one selected configuration for each allowed value of σ_{milli} . In Tables 12–14, for compact presentation, consecutive values assigned to the same configuration are merged into adjacent closed integer ranges. When a configuration changes between two consecutive σ_{milli} values s_i and s_{i+1} , the previous range ends at $\lfloor (s_i + s_{i+1})/2 \rfloor$ and the next range starts at the following integer value. Therefore, the intervals touch, reproduce the selection rule for all allowed σ_{milli} values, and avoid artificial gaps. The implementation identifiers used in the CSV files were converted to the filter-family and parameter notation of Appendix A-B.

ACKNOWLEDGMENT

During the preparation of this study, the author used AI-assisted tools. OpenAI ChatGPT versions 5.2, 5.4, and 5.5 were used to implement the author-designed RDLs-pred research and runtime-benchmarking applications, validation and log-processing scripts, filter-selection model scripts, and to search for related work. The resulting AI-assisted implementation and scripts supported the experiments reported in Section V and summarized in Section VI; AI-assisted literature searches supported Section II. OpenAI ChatGPT versions 5.5 and 5.6 were used for checking and detailed English editing of the manuscript and for translating it into LaTeX. The author designed the study, software, and experiments, reviewed and edited AI-assisted outputs, validated generated code and results through extensive checks, including round-trip, cross-version, checksum-consistency, and log-processing checks, and takes full responsibility for this article.

CONFLICT OF INTEREST

The author declares no conflict of interest. The funder had no role in the design of the study; in the collection, analyses, or interpretation of data; in the writing of the manuscript; or in the decision to submit the article for publication.

REFERENCES

- [1] R. Starosolski, "Application of reversible denoising and lifting steps to DWT in lossless JPEG 2000 for improved bitrates," *Signal Process., Image Commun.*, vol. 39, pp. 249–263, Nov. 2015, doi: [10.1016/j.image.2015.09.013](https://doi.org/10.1016/j.image.2015.09.013).
- [2] R. Starosolski, "Reversible denoising and lifting based color component transformation for lossless image compression," *Multimedia Tools Appl.*, vol. 79, nos. 17–18, pp. 11269–11294, May 2020, doi: [10.1007/s11042-019-08371-w](https://doi.org/10.1007/s11042-019-08371-w).
- [3] R. Starosolski, "Hybrid adaptive lossless image compression based on discrete wavelet transform," *Entropy*, vol. 22, no. 7, p. 751, Jul. 2020, doi: [10.3390/e22070751](https://doi.org/10.3390/e22070751).
- [4] R. Starosolski, "Employing new hybrid adaptive wavelet-based transform and histogram packing to improve JP3D compression of volumetric medical images," *Entropy*, vol. 22, no. 12, p. 1385, Dec. 2020, doi: [10.3390/e22121385](https://doi.org/10.3390/e22121385).
- [5] R. Starosolski, "Application of reversible denoising and lifting steps with step skipping to color space transforms for improved lossless compression," *J. Electron. Imag.*, vol. 25, no. 4, Aug. 2016, Art. no. 043025, doi: [10.1117/1.jei.25.4.043025](https://doi.org/10.1117/1.jei.25.4.043025).
- [6] I. Daubechies and W. Sweldens, "Factoring wavelet transforms into lifting steps," *J. Fourier Anal. Appl.*, vol. 4, no. 3, pp. 247–269, May 1998, doi: [10.1007/bf02476026](https://doi.org/10.1007/bf02476026).
- [7] W. Sweldens, "The lifting scheme: A custom-design construction of biorthogonal wavelets," *Appl. Comput. Harmon. Anal.*, vol. 3, no. 2, pp. 186–200, Apr. 1996, doi: [10.1006/acha.1996.0015](https://doi.org/10.1006/acha.1996.0015).
- [8] H. S. Malvar, G. J. Sullivan, and S. Srinivasan, "Lifting-based reversible color transformations for image compression," *Proc. SPIE*, vol. 7073, Feb. 2008, Art. no. 707307.
- [9] M. Elad, B. Kassar, and G. Vaksman, "Image denoising: The deep learning revolution and beyond—A survey paper," *SIAM J. Imag. Sci.*, vol. 16, no. 3, pp. 1594–1654, Sep. 2023, doi: [10.1137/23m1545859](https://doi.org/10.1137/23m1545859).
- [10] K. Dabov, A. Foi, V. Katkovnik, and K. Egiazarian, "Image denoising by sparse 3-D transform-domain collaborative filtering," *IEEE Trans. Image Process.*, vol. 16, no. 8, pp. 2080–2095, Aug. 2007, doi: [10.1109/TIP.2007.901238](https://doi.org/10.1109/TIP.2007.901238).
- [11] K. Zhang, W. Zuo, Y. Chen, D. Meng, and L. Zhang, "Beyond a Gaussian denoiser: Residual learning of deep CNN for image denoising," *IEEE Trans. Image Process.*, vol. 26, no. 7, pp. 3142–3155, Jul. 2017, doi: [10.1109/TIP.2017.2662206](https://doi.org/10.1109/TIP.2017.2662206).
- [12] S. Guo, Z. Yan, K. Zhang, W. Zuo, and L. Zhang, "Toward convolutional blind denoising of real photographs," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2019, pp. 1712–1722.
- [13] J. Batson and L. Royer, "Noise2Self: Blind denoising by self-supervision," in *Proc. 36th Int. Conf. Mach. Learn.*, in Proceedings of Machine Learning Research, vol. 97, 2019, pp. 524–533.
- [14] Z. Wang, J. Liu, G. Li, and H. Han, "Blind2Unblind: Self-supervised image denoising with visible blind spots," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2022, pp. 2017–2026, doi: [10.1109/CVPR52688.2022.00207](https://doi.org/10.1109/CVPR52688.2022.00207).
- [15] D. Zhang, F. Zhou, Y. Jiang, and Z. Fu, "MM-BSN: Self-supervised image denoising for real-world with multi-mask based on blind-spot network," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. Workshops (CVPRW)*, Jun. 2023, pp. 4189–4198, doi: [10.1109/CVPRW59228.2023.00441](https://doi.org/10.1109/CVPRW59228.2023.00441).
- [16] S. W. Zamir, A. Arora, S. Khan, M. Hayat, F. S. Khan, and M.-H. Yang, "Restormer: Efficient transformer for high-resolution image restoration," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2022, pp. 5718–5729, doi: [10.1109/CVPR52688.2022.00564](https://doi.org/10.1109/CVPR52688.2022.00564).
- [17] D. Zhang and F. Zhou, "Self-supervised image denoising for real-world images with context-aware transformer," *IEEE Access*, vol. 11, pp. 14340–14349, 2023, doi: [10.1109/ACCESS.2023.3243829](https://doi.org/10.1109/ACCESS.2023.3243829).
- [18] T. Plötz and S. Roth, "Benchmarking denoising algorithms with real photographs," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jul. 2017, pp. 2750–2759.
- [19] A. Abdelhamed, S. Lin, and M. S. Brown, "A high-quality denoising dataset for smartphone cameras," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 1692–1700.
- [20] R. Crandall and A. Bilgin, "Lossless image compression using causal block matching and 3D collaborative filtering," in *Proc. IEEE Int. Conf. Image Process. (ICIP)*, Oct. 2014, pp. 5636–5640, doi: [10.1109/ICIP.2014.7026140](https://doi.org/10.1109/ICIP.2014.7026140).
- [21] E. Wige, P. Amon, A. Hutter, and A. Kaup, "In-loop denoising of reference frames for lossless coding of noisy image sequences," in *Proc. IEEE Int. Conf. Image Process.*, Sep. 2010, pp. 461–464, doi: [10.1109/ICIP.2010.5654136](https://doi.org/10.1109/ICIP.2010.5654136).
- [22] D. Lanz, J. Seiler, K. Jaskolka, and A. Kaup, "Compression of dynamic medical CT data using motion compensated wavelet lifting with denoised update," in *Proc. Picture Coding Symp. (PCS)*, Jun. 2018, pp. 56–60.
- [23] D. Lanz, F. Schilling, and A. Kaup, "Scalable lossless coding of dynamic medical CT data using motion compensated wavelet lifting with denoised prediction and update," in *Proc. Picture Coding Symp. (PCS)*, Nov. 2019, pp. 1–5.

- [24] D. Lanz and A. Kaup, "Optimizing rate-distortion performance of motion compensated wavelet lifting with denoised prediction and update," in *Proc. IEEE 22nd Int. Workshop Multimedia Signal Process. (MMSP)*, Sep. 2020, pp. 1–6.
- [25] H. B. Mitchell and D. D. Estrakh, "A fuzzy switched DPCM predictor for lossless compression of noisy images," *Int. J. Uncertainty, Fuzziness Knowl.-Based Syst.*, vol. 5, no. 6, pp. 689–699, Dec. 1997, doi: [10.1142/s0218488597000506](https://doi.org/10.1142/s0218488597000506).
- [26] D. Petrescu, I. Tabus, and M. Gabbouj, "Prediction capabilities of Boolean and stack filters for lossless image compression," *Multidimensional Syst. Signal Process.*, vol. 10, no. 2, pp. 161–187, Apr. 1999, doi: [10.1023/a:1008450429226](https://doi.org/10.1023/a:1008450429226).
- [27] D. D. Estrakh, H. B. Mitchell, P. A. Schaefer, Y. Mann, and Y. Peretz, "'Soft' median adaptive predictor for lossless picture compression," *Signal Process.*, vol. 81, no. 9, pp. 1985–1989, Sep. 2001, doi: [10.1016/s0165-1684\(01\)00058-5](https://doi.org/10.1016/s0165-1684(01)00058-5).
- [28] S. Marusic and G. Deng, "Adaptive prediction for lossless image compression," *Signal Process., Image Commun.*, vol. 17, no. 5, pp. 363–372, May 2002, doi: [10.1016/s0923-5965\(02\)00006-1](https://doi.org/10.1016/s0923-5965(02)00006-1).
- [29] G. Ulacha and M. Łazoryszczak, "Lossless image compression using context-dependent linear prediction based on mean absolute error minimization," *Entropy*, vol. 26, no. 12, p. 1115, Dec. 2024, doi: [10.3390/e26121115](https://doi.org/10.3390/e26121115).
- [30] 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, doi: [10.1109/TCSVT.2012.2221191](https://doi.org/10.1109/TCSVT.2012.2221191).
- [31] *Recommendation H.265: High Efficiency Video Coding*, Standard ISO/IEC 23008-2, ITU-T, Jan. 2026.
- [32] B. Bross, Y.-K. Wang, Y. Ye, S. Liu, J. Chen, G. J. Sullivan, and J.-R. Ohm, "Overview of the versatile video coding (VVC) standard and its applications," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 31, no. 10, pp. 3736–3764, Oct. 2021, doi: [10.1109/TCSVT.2021.3101953](https://doi.org/10.1109/TCSVT.2021.3101953).
- [33] *Recommendation H.266: Versatile Video Coding*, Standard ISO/IEC 23090-3, ITU-T, Jan. 2026.
- [34] S. R. Alvar and F. Kamisli, "On lossless intra coding in HEVC with 3-tap filters," *Signal Process., Image Commun.*, vol. 47, pp. 252–262, Sep. 2016, doi: [10.1016/j.image.2016.06.006](https://doi.org/10.1016/j.image.2016.06.006).
- [35] Y. Liu, Z. Li, Y. Zhang, and R. Zhang, "Lossless compression of infrared images via pixel-adaptive prediction and residual hierarchical decomposition," *Appl. Sci.*, vol. 16, no. 2, p. 1030, Jan. 2026, doi: [10.3390/app16021030](https://doi.org/10.3390/app16021030).
- [36] M. J. Weinberger, G. Seroussi, and G. Sapiro, "The LOCO-I lossless image compression algorithm: Principles and standardization into JPEG-LS," *IEEE Trans. Image Process.*, vol. 9, no. 8, pp. 1309–1324, Aug. 2000, doi: [10.1109/83.855427](https://doi.org/10.1109/83.855427).
- [37] *Information Technology—Lossless and Near-Lossless Compression of Continuous-Tone Still Images: Baseline*, Standard ISO/IEC 14495-1:1999, ISO/IEC and ITU-T, 1999.
- [38] X. Wu and N. Memon, "Context-based, adaptive, lossless image coding," *IEEE Trans. Commun.*, vol. 45, no. 4, pp. 437–444, Apr. 1997, doi: [10.1109/26.585919](https://doi.org/10.1109/26.585919).
- [39] I. Pitas and A. N. Venetsanopoulos, "Nonlinear order statistic filters for image filtering and edge detection," *Signal Process.*, vol. 10, no. 4, pp. 395–413, Jun. 1986, doi: [10.1016/0165-1684\(86\)90047-2](https://doi.org/10.1016/0165-1684(86)90047-2).
- [40] D. R. K. Brownrigg, "The weighted median filter," *Commun. ACM*, vol. 27, no. 8, pp. 807–818, Aug. 1984, doi: [10.1145/358198.358222](https://doi.org/10.1145/358198.358222).
- [41] L. Yin, R. Yang, M. Gabbouj, and Y. Neuvo, "Weighted median filters: A tutorial," *IEEE Trans. Circuits Syst. II, Analog Digit. Signal Process.*, vol. 43, no. 3, pp. 157–192, Mar. 1996, doi: [10.1109/82.486465](https://doi.org/10.1109/82.486465).
- [42] Y. Dong and S. Xu, "A new directional weighted median filter for removal of random-valued impulse noise," *IEEE Signal Process. Lett.*, vol. 14, no. 3, pp. 193–196, Mar. 2007, doi: [10.1109/LSP.2006.884014](https://doi.org/10.1109/LSP.2006.884014).
- [43] C. Tomasi and R. Manduchi, "Bilateral filtering for gray and color images," in *Proc. 6th Int. Conf. Comput. Vis.*, Apr. 1998, pp. 839–846.
- [44] K. He, J. Sun, and X. Tang, "Guided image filtering," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 35, no. 6, pp. 1397–1409, Jun. 2013, doi: [10.1109/TPAMI.2012.213](https://doi.org/10.1109/TPAMI.2012.213).
- [45] X. Z. Sun and A. N. Venetsanopoulos, "Adaptive schemes for noise filtering and edge detection by use of local statistics," *IEEE Trans. Circuits Syst.*, vol. 35, no. 1, pp. 57–69, Jan. 1988, doi: [10.1109/31.1700](https://doi.org/10.1109/31.1700).
- [46] A. Buades, B. Coll, and J.-M. Morel, "A non-local algorithm for image denoising," in *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2005, pp. 60–65.
- [47] D. Taubman, "High performance scalable image compression with EBCOT," *IEEE Trans. Image Process.*, vol. 9, no. 7, pp. 1158–1170, Jul. 2000, doi: [10.1109/83.847830](https://doi.org/10.1109/83.847830).
- [48] A. J. Pinho, "On the impact of histogram sparseness on some lossless image compression techniques," in *Proc. Int. Conf. Image Process.*, vol. 3, Oct. 2001, pp. 442–445.
- [49] R. Starosolski, "Compressing high bit depth images of sparse histograms," in *AIP Conf. Proc.*, 2008, pp. 269–272, doi: [10.1063/1.3037069](https://doi.org/10.1063/1.3037069).
- [50] (2009). *The Waterloo Greyscale Set 2*. Accessed: May 30, 2026. [Online]. Available: <https://links.uwaterloo.ca/Repository.html>
- [51] R. Starosolski. (2016). *Image Transformation Tools: Colortransf Version 1.1 and Pgmtool Version 0.73b*. Accessed: Jul. 31, 2026. [Online]. Available: <https://sun.aei.polsl.pl/~rstaros/imgtransf/index.html>
- [52] *Sets A1, A2, and A3 of Color Images*. Accessed: May 30, 2026. [Online]. Available: <https://doi.org/10.6084/m9.figshare.1396485>
- [53] *Information Technology—Lossless and Near-Lossless Compression of Continuous-Tone Still Images: Extensions*, Standard ISO/IEC 14495-2:2003, ISO/IEC and ITU-T, 2003.
- [54] D. L. Donoho and I. M. Johnstone, "Ideal spatial adaptation by wavelet shrinkage," *Biometrika*, vol. 81, no. 3, pp. 425–455, Sep. 1994.
- [55] OpenCV. (2025). *The OpenCV Implementation, Version 4.12.0*. Accessed: May 14, 2026. [Online]. Available: <https://github.com/opencv/opencv>
- [56] Kakadu Software. (2025). *The Kakadu JPEG 2000 Implementation, Version 8.6.2*. Accessed: Mar. 21, 2026. [Online]. Available: <https://kakadusoftware.com/>
- [57] A. Skodras, C. Christopoulos, and T. Ebrahimi, "The JPEG 2000 still image compression standard," *IEEE Signal Process. Mag.*, vol. 18, no. 5, pp. 36–58, Sep. 2001, doi: [10.1109/79.952804](https://doi.org/10.1109/79.952804).
- [58] *Information Technology—JPEG 2000 Image Coding System—Part 1: Core Coding System*, Standard ISO/IEC 15444-1:2024, ISO/IEC, 2024.
- [59] J. Sneyers, J. Alakuijala, L. Versari, Z. Szabadka, S. Boukourt, A. Cohen-Tidhar, M. Firsching, E. Kliuchnikov, T. Lev-Ami, E. Portis, T. Richter, and O. Watanabe, "The JPEG XL image coding system: History, features, coding tools, design rationale, and future," 2025, *arXiv:2506.05987*.
- [60] *Information Technology—JPEG XL Image Coding System—Part 1: Core Coding System*, Standard ISO/IEC 18181-1:2024, ISO/IEC, 2024.
- [61] libjxl. (2026). *The Libjxl JPEG XL Implementation, Version 0.11.2*. Accessed: May 14, 2026. [Online]. Available: <https://github.com/libjxl/libjxl/releases>
- [62] CharLS. (2026). *The CharLS JPEG-LS Implementation, Version 2.4.3*. Accessed: May 14, 2026. [Online]. Available: <https://github.com/team-charls/charls/releases>
- [63] OpenJPEG. (2025). *The OpenJPEG JPEG 2000 Implementation, Version 2.5.4*. Accessed: May 30, 2026. [Online]. Available: <https://github.com/uclouvain/openjpeg>



ROMAN STAROSOLSKI received the Ph.D. and D.Sc. degrees in computer science from the Silesian University of Technology, Gliwice, Poland, in 2002 and 2017, respectively. He is currently an Associate Professor and the Deputy Head of the Department of Algorithms and Software, Faculty of Automatic Control, Electronics and Computer Science, Silesian University of Technology. He has authored about 35 journal articles and conference papers and holds eight patents. His research interests include image and data compression, biomedical imaging, image-coding standards, lifting-based reversible transforms, and adaptive algorithms.

...