Journal of Electronic Imaging

JElectronicImaging.org

Application of reversible denoising and lifting steps with step skipping to color space transforms for improved lossless compression

Roman Starosolski

SPIE. IS&T imaging.org

Roman Starosolski, "Application of reversible denoising and lifting steps with step skipping to color space transforms for improved lossless compression," *J. Electron. Imaging* **25**(4), 043025 (2016), doi: 10.1117/1.JEI.25.4.043025.

Application of reversible denoising and lifting steps with step skipping to color space transforms for improved lossless compression

Roman Starosolski*
Silesian University of Technology, Institute of Informatics, Akademicka 16, 44-100 Gliwice, Poland

Abstract. Reversible denoising and lifting steps (RDLS) are lifting steps integrated with denoising filters in such a way that, despite the inherently irreversible nature of denoising, they are perfectly reversible. We investigated the application of RDLS to reversible color space transforms: RCT, YCoCg-R, RDgDb, and LDgEb. In order to improve RDLS effects, we propose a heuristic for image-adaptive denoising filter selection, a fast estimator of the compressed image bitrate, and a special filter that may result in skipping of the steps. We analyzed the properties of the presented methods, paying special attention to their usefulness from a practical standpoint. For a diverse image test-set and lossless JPEG-LS, JPEG 2000, and JPEG XR algorithms, RDLS improves the bitrates of all the examined transforms. The most interesting results were obtained for an estimation-based heuristic filter selection out of a set of seven filters; the cost of this variant was similar to or lower than the transform cost, and it improved the average lossless JPEG 2000 bitrates by 2.65% for RDgDb and by over 1% for other transforms; bitrates of certain images were improved to a significantly greater extent. [DOI: 10.1117/1.JEI.25.4.043025]

Keywords: lossless image compression; lifting technique; reversible color space transform; denoising; reversible denoising and lifting step.

Paper 16307 received Apr. 14, 2016; accepted for publication Jul. 22, 2016; published online Aug. 12, 2016; corrected Aug. 27, 2019.

1 Introduction

Most color image compression algorithms independently compress the image components; since components in the RGB space are correlated, the compression is performed after transforming image data to a less correlated color space. For the lossless compression, reversible color space transforms are employed, which are built using lifting steps.[1,2] In Ref. 3, it was noticed that such a step might increase the total amount of noise that had to be encoded during compression. To remove correlation without increasing noise, lifting steps were replaced with reversible denoising and lifting steps (RDLS), which are lifting steps integrated with denoising filters. The step is modified in such a way that, despite involving the inherently irreversible denoising, it is perfectly reversible. RDLS was applied to a simple RDgDb[4] transform (also known as $A_{2,1}$[5]) and it was found that RDLS improved bitrates of images in optical resolutions of acquisition devices. Experiments were performed for three significantly different standard image compression algorithms in the lossless mode (JPEG-LS,[6] JPEG 2000,[7] and JPEG XR[8]) and for simple linear denoising filters ("smoothing" filters). The memoryless entropy of the component prediction error obtained with the MED predictor[9] was a very efficient estimator of image component transform effects that was found suitable for selecting a filter for a given image component independently of the image compression algorithm.

*Address all correspondence to: Roman Starosolski, E-mail: roman.starosolski@polsl.pl

An intermediate stage of the research reported herein was presented in Ref. 10, where RDLS was applied to more complex color space transforms LDgEb[4] (denoted $A_{4,10}$ in Ref. 5) and RCT (among others, used in JPEG 2000). RDLS effects were evaluated using the same denoising filters, compression algorithms, and test images, as in Ref. 3. Entropy estimation employing MED was used for selecting the denoising filter and deciding whether to exploit denoising. The selection of filters for a given transform step was based only on the estimated filtering effects on a bitrate of component modified by this step. As a result, the filtering might, for RCT and LDgEb, result in worsening of the overall image bitrate even if assuming the perfect estimation.

In Ref. 3, it was also observed that, although RDgDb or its RDLS-modified variant improves bitrates in the average case, for some color images, the best ratios were obtained when untransformed components were compressed. In Ref. 11, RDLS was applied to discrete wavelet transform (DWT) in lossless JPEG 2000 compression of grayscale images. The noise filtering was the most effective when applied only to some steps. Some images were compressed better when the DWT stage of this algorithm was skipped, although in the average case, RDLS improved bitrates. Thus, it was suspected that the optimum might be in-between skipping and applying the transform, i.e., that better bitrates may be obtained by skipping only some of the steps of the transform.

The new contributions of this study are mainly motivated by conclusions from earlier works and are aimed at properties of RDLS-modified color space transforms that are worthwhile from a practical standpoint. Since sometimes it is better to compress an untransformed image, we propose

employing a special filter, named "null," which may make the RDLS-modified color space transform skip all or some of its steps. We also propose an image-adaptive denoising filter selection heuristic that, as compared to the heuristic exploited in Ref. 10, avoids worsening of the image bitrate and for more complex transforms is faster. As the heuristic cost (i.e., its computational time complexity) may still be too high for practical applications, we propose the limited complexity estimator of compression effects H0_pMED(10k:100), which is based on a fast estimator from Ref. 5. The cost of the latter, however, significantly increased if RDLS was employed. The evaluation of the effects of the above contributions is performed using the same transforms, compression algorithms, and test images, as in Ref. 10. Additionally, we apply RDLS to the YCoCg-R transform,[1] test the heuristic against an exhaustive filter search, and the estimators against the actual bitrates of lossless image compression algorithms.

The remainder of this paper is organized as follows. In Sec. 2, first, the reversible color space transforms and the RDLS method are briefly described. Next, in Sec. 2.3, we present the RDLS-modified transforms including the new RDLS-YCoCg-R and compare their dynamic ranges and bit-depths to the non-RDLS counterparts. In Sec. 2.4, we demonstrate the transform reversibility in a step-by-step example and present sample effects that RDLS may have on the transformed components of a noisy image. In Sec. 2.5, we describe the denoising filters used in the research including the proposed null filter. Then we introduce the filter selection heuristic (Sec. 2.6) along with the compression effect estimators including the proposed H0_pMED(10k:100) (Sec. 2.7) and report their complexities for various transforms. We also describe the compression algorithms, implementations, and test data (Sec. 2.8). Results are presented and discussed in Sec. 3. We start from analyzing the effects of contributions that are aimed at the bitrate improvement (i.e., the new heuristic and the null filter) and comparing them to the previously known methods. Next (Sec. 3.2), we reduce the cost of the bitrate improvement by limiting the number of denoising filters and iterations of the heuristic and by applying H0_pMED(10k:100). We also perform certain additional experiments (Sec. 3.3), among others, to check how far from optimal are our heuristic and estimation method. Finally, the extensive summary is followed by a brief conclusion.

2 Materials and Methods

2.1 *Lifting-Based Reversible Color Space Transforms*

Reversible color space transforms investigated in this study are sequences of lifting steps. Below, we characterize them briefly and refer the reader to Refs. 1, 4, and 5 for more detailed descriptions and comparisons among them. In each lifting step of a transform, a single pixel component is modified by adding to it a linear combination of other components of the same pixel; the sum may be negated. Up to three steps are needed to transform an RGB pixel to any of the spaces discussed in this section. A transform realized as a sequence of lifting steps has advantageous properties: it may be computed in-place, it is reversible when transformed components are stored using integers (it is integer-reversible), and it is easily and perfectly invertible. To obtain an inverse transform, the inverses of lifting steps should be applied in an order exactly opposite to the order employed by the forward transform. However, transforms are usually presented using different symbols for components before and after applying lifting steps to them, lifting equations are simplified, or a lifting sequence is transformed in order to present an optimized method of transform implementation. Thus, it may not be obvious which component is modified in a given step or how to inverse the step. In this section, we follow the usual way of presentation; for transforms presented as sequences of lifting steps, see Sec. 2.3.

Probably the most frequently used reversible color space transform is the RCT transform employed in JPEG 2000 for lossless compression, which is an approximation of an irreversible ICT transform used in JPEG 2000 for lossy compression, that in turn may be seen as an approximation of an irreversible YCbCr color space transform.[7] Equation (1) presents forward (left-hand side) and inverse RCT:

$$\begin{aligned} Yr &= \lfloor (R+2G+B)/4 \rfloor & & G = Yr - \lfloor (Ur+Vr)/4 \rfloor \\ Ur &= R-G & \Leftrightarrow\ & R = Ur+G \\ Vr &= B-G & & B = Vr+G \end{aligned}, \tag{1}$$

where the $\lfloor i/2^q \rfloor$ is the floor of $i/2^q$; i.e., the greatest integer not exceeding $i/2^q$, that for integer i and positive integer q may be simply computed as the arithmetic right shift of i by q bits. RCT transforms RGB primary color components R, G, and B to component Yr representing pixel luminance and two chrominance components Ur and Vr. Note that compared to primary color and luminance components, the dynamic range of chrominance components and consequently their bit-depths are greater—which is also true for other transforms presented in this section.

Another standard transform, among others used in JPEG XR, is YCoCg-R. In Eq. (2), it is presented as it was originally proposed in Ref. 12, i.e., as a sequence of steps involving storing an intermediate result in a temporary variable t; Y represents pixel luminance, whereas Co and Cg are chrominance components:

$$\begin{aligned} Co &= R-B & & t = Y - \lfloor Cg/2 \rfloor \\ t &= B + \lfloor Co/2 \rfloor & \Leftrightarrow\ & G = Cg + t \\ Cg &= G - t & & B = t - \lfloor Co/2 \rfloor \\ Y &= t + \lfloor Cg/2 \rfloor & & R = B + Co \end{aligned}. \tag{2}$$

In Ref. 3, RDLS was applied to the RDgDb transform [Eq. (3)],[4] which is also known as $A_{2,1}$.[5] RDgDb was chosen because of its simplicity and good performance. It requires only two simple integer operations (add or subtract) per pixel, which for the three-component RGB color space is possible because we do not transform all the components. There are two transformed chrominance components Dg and Db, but instead of luminance, the untransformed primary color R is used:

$$\begin{aligned} R &= R & & R = R \\ Dg &= R-G & \Leftrightarrow\ & G = R - Dg \\ Db &= G-B & & B = G - Db \end{aligned}. \tag{3}$$

In this research, we also include the LDgEb transform [Eq. (4)][4] (denoted $A_{4,10}$ in Ref. 5). Like typical transforms,

it results in a luminance (L) and two chrominance (Dg and Eb) components; interestingly, the component formulas are based on actual analog transforms from the human visual system:

$$\begin{array}{l} Dg = R - G \\ L = R - \lfloor Dg/2 \rfloor \\ Eb = B - L \end{array} \quad \Leftrightarrow \quad \begin{array}{l} R = L + \lfloor Dg/2 \rfloor \\ G = R - Dg \\ B = Eb + L \end{array} . \tag{4}$$

2.2 *Reversible Denoising and Lifting Step*

A lifting step in a reversible color space transform may propagate the noise to the component it modifies from other components. In Ref. 3, integrating denoising into lifting steps was proposed in order to avoid noise propagation while preserving other transform properties (i.e., reversibility, in-place operation, and removing correlation). The method was based on the generalized lifting step of a color space transform:

$$C_x \leftarrow C_x \oplus \mathrm{f}(C_1, \ldots, C_{x-1}, C_{x+1}, \ldots, C_m), \tag{5}$$

where C_n is the n'th component of the pixel, C_x is the component which is modified by the step, m is the number of components, f is a deterministic function, and the operation $\oplus$ is reversible, i.e., an inverse operation $\otimes$ exists, such that $c = a \oplus b \Leftrightarrow a = c \otimes b$.

By denoising of arguments of function f in the generalized lifting step [Eq. (5)], a reversible denoising and lifting step [RDLS, Eq. (6)] was constructed:

$$C_x \leftarrow C_x \oplus \mathrm{f}(C_1^d, \ldots, C_{x-1}^d, C_{x+1}^d, \ldots, C_m^d), \tag{6}$$

where C_n^d is the denoised n'th component of the pixel. Different denoising filters may be employed for different components and in different steps. For denoising of arguments of function f, any component of any pixel may be used except the C_x of the pixel to which the RDLS is being applied. Denoising is not an in-place operation, computing the function f argument C_n^d does not alter C_n. In this study, for denoising of C_n of a specific pixel, we use C_n of this pixel and of its neighbors.

Despite the inherently lossy nature of denoising, RDLS exploiting denoising is perfectly and easily invertible. An inverse of an RDLS-modified color space transform is obtained by applying inverses of RDLS:

$$C_x \leftarrow C_x \otimes \mathrm{f}(C_1^d, \ldots, C_{x-1}^d, C_{x+1}^d, \ldots, C_m^d), \tag{7}$$

in an order opposite to one employed by the forward transform—see examples in the following two sections. Naturally, the same denoising filters must be used for the same components in inverse RDLS, as they were applied during forward RDLS.

2.3 *Reversible Denoising and Lifting Steps-Modified Transforms*

In Eq. (8), the RCT transform is defined as a sequence of lifting steps—both the forward RCT transform (left-hand side) and inverse:

$$\begin{array}{ll} \text{step 1:} & C_1 \leftarrow C_1 - C_2 \\ \text{step 2:} & C_3 \leftarrow C_3 - C_2 \\ \text{step 3:} & C_2 \leftarrow C_2 + \lfloor (C_1 + C_3)/4 \rfloor \end{array} \quad \Leftrightarrow \quad \begin{array}{ll} \text{step 1:} & C_2 \leftarrow C_2 - \lfloor (C_1 + C_3)/4 \rfloor \\ \text{step 2:} & C_3 \leftarrow C_3 + C_2 \\ \text{step 3:} & C_1 \leftarrow C_1 + C_2 \end{array} . \tag{8}$$

We use the notation as in Eqs. (5)–(7), where the same symbol denotes the pixel's component before and after modifying it by the lifting step or the RDLS. For all the transforms presented in this section, C_1, C_2, and C_3 denote the R, G, and B components of the untransformed image, respectively. For RCT, the C_1, C_2, and C_3 denote also the Ur, Yr, and Vr components of the transformed image, respectively. Generally, the steps must be performed in a specified order.

The RDLS-modified RCT [RDLS-RCT, Eq. (9)] is obtained by simply replacing the RCT [Eq. (8)] lifting steps [Eq. (5)] with the RDLS [Eq. (6)] constructed based on them:

$$\begin{array}{ll} \text{step 1:} & C_1 \leftarrow C_1 - C_2^d \\ \text{step 2:} & C_3 \leftarrow C_3 - C_2^d \\ \text{step 3:} & C_2 \leftarrow C_2 + \lfloor (C_1^d + C_3^d)/4 \rfloor \end{array} \quad \Leftrightarrow \quad \begin{array}{ll} \text{step 1:} & C_2 \leftarrow C_2 - \lfloor (C_1^d + C_3^d)/4 \rfloor \\ \text{step 2:} & C_3 \leftarrow C_3 + C_2^d \\ \text{step 3:} & C_1 \leftarrow C_1 + C_2^d \end{array} . \tag{9}$$

We use the same symbols to denote components of regular transforms and of their RDLS-modified counterparts; thus, C_1, C_2, and C_3 denote the Ur, Yr, and Vr components of the RDLS-RCT transformed image, respectively. The regular lifting transform is a special case of the RDLS-modified transform; the lifting transform may be obtained by using a denoising filter, for which $C_n^d = C_n$. We call such a filter the "none" filter.

The dynamic range of RDLS-RCT components differs from RCT components' range in the case of the C_2 component. We assume that denoising of the pixel component C_n may result in any integer within the dynamic range of the image component C_n. Note that the "component" term may refer to a pixel and to an image; in the latter case, the image component C_n is an image consisting of C_n components of all pixels of a color image. In RDLS-RCT, for the $[0, 2^b - 1]$ range of untransformed RGB components, the range of C_1 and C_3 before performing the forward step 3 is $[-2^b + 1, 2^b - 1]$. Due to denoising, step 3 of RDLS-RCT adds to C_2, a floor of a quarter of a sum of two values, each of which may be any integer from the $[-2^b + 1, 2^b - 1]$ range. Thus, the range of the RDLS-RCT transformed C_2 is $[-2^{b-1}, 3 \cdot 2^{b-1} - 2]$.

As noted in Ref. 3, a lifting-based color space transform may be performed for each pixel independently of others. The RDLS sequence, constructed based on a color space transform, is a transform of the whole image components.

It is not a color space transform, since denoising of a specific pixel's component C_n requires to access the C_n of (at least) neighboring pixels. The lifting-based color space transform of an image may be performed in a pixel-by-pixel regime, i.e., we apply all transform steps to a pixel, then we proceed to the next pixel, or step-by-step, i.e., we apply a lifting step to all pixels, and then we proceed to the next step; these regimes are equivalent. Also for the RDLS-modified transform, both regimes may be exploited, but they are not equivalent as filters depend on the regime. In this study, we employ the step-by-step regime. In this regime, for each image component, except for the component being modified in the current step, either all pixels are transformed or all are untransformed. Assuming that each untransformed and each transformed image component has invariant characteristics, the same filter may be selected for all image pixels in a given RDLS-modified transform step for denoising of a given component.

Presented below are the RDLS-modified variants of YCoCg-R [RDLS-YCoCg-R, Eq. (10)], RDgDb [RDLS-RDgDb, Eq. (11)], and LDgEb [RDLS-LDgEb, Eq. (12)]:

$$\begin{array}{ll} \text{step 1:} & C_1 \leftarrow C_1 - C_3^d \\ \text{step 2:} & C_3 \leftarrow -C_3 - \lfloor C_1^d/2 \rfloor + C_2^d \\ \text{step 3:} & C_2 \leftarrow C_2 - \lceil C_3^d/2 \rceil \end{array} \Leftrightarrow \begin{array}{ll} \text{step 1:} & C_2 \leftarrow C_2 + \lceil C_3^d/2 \rceil \\ \text{step 2:} & C_3 \leftarrow -C_3 - \lfloor C_1^d/2 \rfloor + C_2^d, \\ \text{step 3:} & C_1 \leftarrow C_1 + C_3^d \end{array} \tag{10}$$

$$\begin{array}{ll} \text{step 1:} & C_3 \leftarrow -C_3 + C_2^d \\ \text{step 2:} & C_2 \leftarrow -C_2 + C_1^d \\ \text{step 3:} & C_1 \leftarrow C_1 \end{array} \Leftrightarrow \begin{array}{ll} \text{step 1:} & C_1 \leftarrow C_1 \\ \text{step 2:} & C_2 \leftarrow -C_2 + C_1^d, \\ \text{step 3:} & C_3 \leftarrow -C_3 + C_2^d \end{array} \tag{11}$$

$$\begin{array}{ll} \text{step 1:} & C_2 \leftarrow -C_2 + C_1^d \\ \text{step 2:} & C_1 \leftarrow C_1 - \lfloor C_2^d/2 \rfloor \\ \text{step 3:} & C_3 \leftarrow C_3 - C_1^d \end{array} \Leftrightarrow \begin{array}{ll} \text{step 1:} & C_3 \leftarrow C_3 + C_1^d \\ \text{step 2:} & C_1 \leftarrow C_1 + \lfloor C_2^d/2 \rfloor. \\ \text{step 3:} & C_2 \leftarrow -C_2 + C_1^d \end{array} \tag{12}$$

By applying to the RDLS-modified transforms the none filter, the non-RDLS variants may be obtained in a form of sequences of lifting steps. Note that in Eq. (10), the $\lceil C_3^d/2 \rceil$ is the smallest integer greater or equal to $C_3^d/2$. Names of the transformed components are, for each transform, listed in Table 1.

Table 1 also presents dynamic ranges and bit-depths of components of all transforms investigated in this study. As opposed to RDLS-RDgDb, some components of RDLS-RCT (Yr), RDLS-YCoCg-R (Y and Cg), and RDLS-LDgEb (L and Eb) may require greater bit-depths than their non-RDLS counterparts. In such cases, the range of the non-RDLS transformed component is placed approximately in the center of the corresponding RDLS transformed component range. The dynamic range expansion with respect to the non-RDLS transform is the greatest for the RDLS-YCoCg-R transform, which needs 2 bits more than YCoCg-R for encoding the Y component.

2.4 *Example of a Reversible Denoising and Lifting Steps-Modified Color Space Transform*

In this section, using the RDLS-RCT as an example, we demonstrate how an RDLS-modified transform processes an image and how the transform reversibility is maintained despite involving the inherently irreversible denoising. Next, we present sample effects that RDLS may have on the transformed components.

The diagram in Fig. 1 presents operations performed by consecutive steps of forward and inverse RDLS-RCT. Effects of these operations on components of a sample noisy image are presented in Fig. 2; letters surrounded by dashed lines in Fig. 1 denote the panels in Fig. 2 that contain the current

Table 1 Names, dynamic ranges and bit-depths of components of lifting transforms and their RDLS counterparts; b, bit-depth of the untransformed RGB components, $b > 1$.

	C_1			C_2			C_3		
Transform	Name	range	depth	name	range	depth	name	range	depth
RGB	R	$[0, 2^b - 1]$	b	G	$[0, 2^b - 1]$	b	B	$[0, 2^b - 1]$	b
RCT	Ur	$[-2^b + 1, 2^b - 1]$	$b+1$	Yr	$[0, 2^b - 1]$	b	Vr	$[-2^b + 1, 2^b - 1]$	$b+1$
RDLS-RCT	Ur	$[-2^b + 1, 2^b - 1]$	$b+1$	Yr	$[-2^{b-1}, 3 \cdot 2^{b-1} - 2]$	$b+1$	Vr	$[-2^b + 1, 2^b - 1]$	$b+1$
YCoCg-R	Co	$[-2^b + 1, 2^b - 1]$	$b+1$	Y	$[0, 2^b - 1]$	b	Cg	$[-2^b + 1, 2^b - 1]$	$b+1$
RDLS-YCoCg-R	Co	$[-2^b + 1, 2^b - 1]$	$b+1$	Y	$[-3 \cdot 2^{b-2}, 7 \cdot 2^{b-2} - 2]$	$b+2$	Cg	$[-3 \cdot 2^{b-1} + 2, 3 \cdot 2^{b-1} - 1]$	$b+2$
RDgDb	R	$[0, 2^b - 1]$	b	Dg	$[-2^b + 1, 2^b - 1]$	$b+1$	Db	$[-2^b + 1, 2^b - 1]$	$b+1$
RDLS-RDgDb	R	$[0, 2^b - 1]$	b	Dg	$[-2^b + 1, 2^b - 1]$	$b+1$	Db	$[-2^b + 1, 2^b - 1]$	$b+1$
LDgEb	L	$[0, 2^b - 1]$	b	Dg	$[-2^b + 1, 2^b - 1]$	$b+1$	Eb	$[-2^b + 1, 2^b - 1]$	$b+1$
RDLS-LDgEb	L	$[-2^{b-1} + 1, 3 \cdot 2^{b-1} - 1]$	$b+1$	Dg	$[-2^b + 1, 2^b - 1]$	$b+1$	Eb	$[-3 \cdot 2^{b-1} + 1, 3 \cdot 2^{b-1} - 2]$	$b+2$

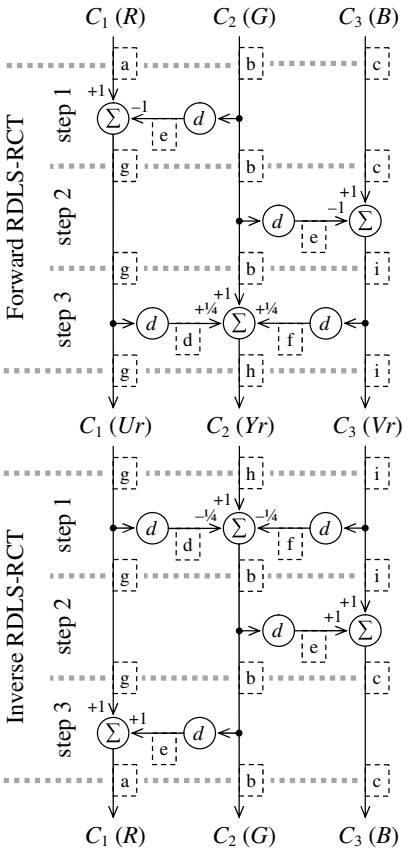

Fig. 1 Example of forward and inverse RDLS-RCT. $\sum$, weighted sum of components; *d*, denoising of a component; dashed lines surround labels of Fig. 2 panels with the transformed component of a sample image.

appearance of the component. In this example, we use the same denoising filter in all transform steps for all components requiring denoising and employ the step-by-step regime.

Step 1 of forward RDLS-RCT applied to all image pixels transforms C_1, C_2, and C_3 components of the untransformed image (i.e., R, G, and B primary color components, respectively) by modifying only the C_1 component. The latter is modified by subtracting from it the temporarily created component obtained by denoising of the C_2 component. In step 2, we modify in an analogical way the C_3 component. Step 3 ($C_2 \leftarrow C_2 + \lfloor (C_1^d + C_3^d)/4 \rfloor$) is more complicated. For brevity in Fig. 1, we ignore computing of the floor. In this step, we add to the C_2 component the quarter of the sum of the temporary denoised components C_1 and C_3, which are obtained based on the components already transformed in earlier steps. To C_2 of each pixel, we add the quarter of the sum of the temporary denoised components C_1 and C_3 of the same pixel. After step 3, we have the RDLS-RCT transformed C_1, C_2, and C_3 components, i.e., Ur, Yr, and Vr, respectively.

Inverse transform simply applies inverses of the forward transform steps in a reversed order. Step 1 of the inverse transform ($C_2 \leftarrow C_2 - \lfloor (C_1 + C_3)/4 \rfloor$) is an inverse of the forward step 3. These two steps modify the C_2 component only. Other components are not changed by them but are used in a read-only manner to obtain temporary denoised components. Thus, components C_1 and C_3 before inverse transform step 1 are exactly the same as before forward step 3. Inverse transform step 1 subtracts from C_2 the quarter of the sum of the denoised components C_1 and C_3—subtracts from C_2 of each pixel exactly the same value that was added to it in forward step 3. The reversibility is maintained because, based on the same C_1 and C_3 as in forward step 3, we obtained the same temporary denoised components. Therefore, the denoising filter must be deterministic but does not have to be reversible, or invertible; we perform "forward" denoising in both the forward and the inverse RDLS-modified color space transform. In this example, we use the same denoising filter for all the components. However, in a general case, for denoising of a given component in the inverse of a certain forward step, we must use the same denoising filter that was used for this component in the forward step. As a result of inverse step 1, we obtain the untransformed C_2, i.e., the G primary color component, that is then used in inverse transform steps 2 and 3 to obtain the temporary denoised untransformed C_2. The latter was in forward transform subtracted from the untransformed C_1 and C_3; in inverse transform, we add it to the transformed C_3 and C_1, reconstructing the untransformed C_3 and C_1—the primary color components B and R.

In Fig. 2, we compare effects of RCT and RDLS-RCT for a noisy image. Components of an original untransformed image [Figs. 2(a)–2(c)] are contaminated with impulse noise (10% of white pixels were replaced by black ones). Impulse noise is not a typical distortion found in real-life images; we use it because it is easy to observe and, in most cases, may be efficiently removed by using a simple median denoising filter. Hence, in this example, for denoising, within all RDLS steps of RDLS-RCT we employ the median denoising filter with 3×3 pixels window. This filter may fail to remove noise from components of our image, or introduce distortions, only at the edges between areas of different brightness [in Fig. 2, compare panels (d), (e), and (f) with (g), (b), and (i), respectively]. Figure 2 also reports the component bit-depths and bitrates of a lossless image compression algorithm (estimated using the H0_pMED estimator, which is described in Sec. 2.7).

Looking at the effects of RCT [Figs. 2(j)–2(l)], we see that the transformed components contain noise from all the components used to calculate them. When we compress these components independently, then we encode the information on noise from the untransformed components twice (noise from the C_1 and C_3 components) or three times (noise from C_2). The most pronounced effect of RDLS [Figs. 2(g)–2(i)] is that the transformed components contain noise only from their untransformed counterparts.

A more subtle difference between RDLS-based and lifting-based transform effects may be noticed for components that, during transform, are modified based on themselves. For example, the component C_2 in step 3 is modified based on components C_1 and C_3 that have already been modified based on C_2 in earlier steps 1 and 2, respectively. Basically, step 3 of the lifting-based RCT transform makes C_2 contain a weighted arithmetic mean of the untransformed C_1, C_2, and C_3 components. Therefore, it decreases in C_2 the amplitude of the signal originally present in this component, that is, of both the ideal noise-free image and noise contaminating it. On the other hand, assuming the perfect denoising, the RDLS modifies a component based only on the ideal

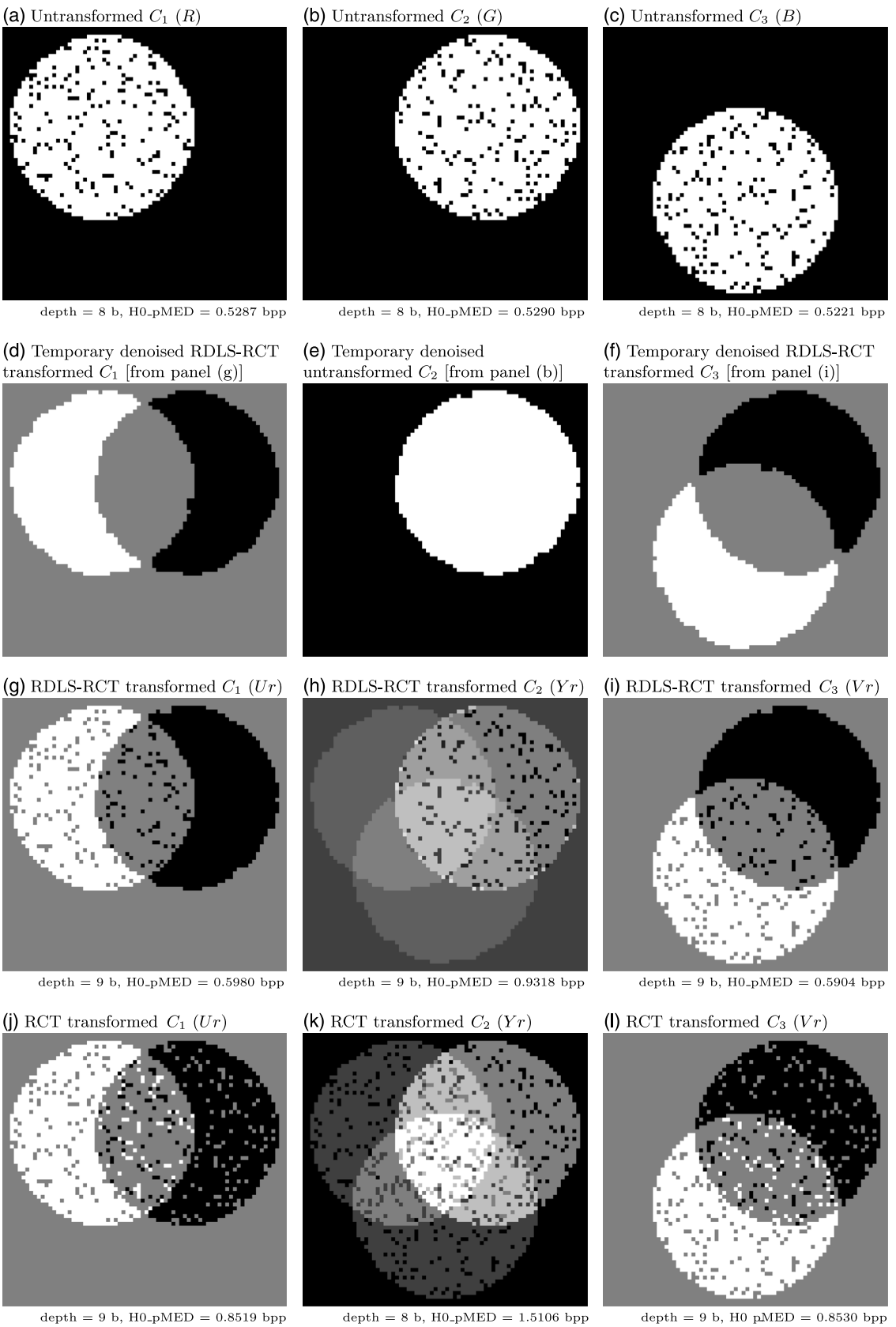

Fig. 2 Effects of RCT and RDLS-RCT on components of a noisy image. (a–c) Untransformed components of the original image, (d–f) temporary denoised components created while computing RDLS-RCT, (g–i) RDLS-RCT transformed components, and (j–l) RCT transformed components; image sizes are 74×71 pixels, transformed components are presented normalized to the dynamic range of original ones, for denoising the median filter with 3×3 pixels window was used; H0_pMED, estimated bitrate of the component (see Sec. 2.7).

noise-free images contained in other components. Thus, the ideal noise-free image is transformed differently than noise. The former is transformed as in a regular lifting-based transform, while the latter does not get propagated to other components and cannot be "weighted" using its copy from other components. The RDLS-RCT transformed C_2 contains a weighted arithmetic mean of ideal noise-free images from the C_1, C_2, and C_3 components and unmodified noise originally present in C_2. Therefore, the amplitude of noise from unmodified C_2 in transformed C_2 is greater after RDLS-RCT than after RCT. The above effect may be hard to notice on Fig. 2. The pixels affected by noise from untransformed C_2 appear similarly dark in C_2 transformed by RCT [Fig. 2(k)] and RDLS-RCT [Fig. 2(h)] because all components in Fig. 2 are presented normalized to the dynamic range of untransformed primary colors, so the actual dynamic range of C_2 in the case of RDLS-RCT was reduced twice, whereas for RCT, it was not changed.

It is worth noting that the actual denoising is not perfect, which may affect bitrates of RDLS-modified color space transforms. For our example image, the effects of the imperfect denoising we applied are rough edges between areas of different brightness (noticeable in all RDLS-RCT transformed components) and additional noisy pixels in transformed C_2 (of intensity different to pixels that were noisy in untransformed C_2).

The estimated bitrates of RDLS-RCT components are significantly lower than bitrates of RCT components. Interestingly, bitrates of untransformed components are even better, which suggests that for some images, the untransformed components should be compressed. In the next section, we propose a special case of a denoising filter for RDLS, which may result in skipping of the RDLS-modified color space transform.

2.5 *Denoising Filters*

For denoising, we employed simple low-pass linear averaging filters (smoothing filters) with 3×3 pixels windows; these filters were previously found effective for RDLS-RDgDb, RDLS-LDgEb, and RDLS-RCT.[3,10] The filtered pixel component C_n^d was calculated as a weighted arithmetic mean of the C_n components of pixels from the window. The weight of the window center point varied for different filters, while its neighbors' weights were fixed to 1. We tested 11 smoothing filters with window center point weights from 1 to 1024 (integer powers of 2 only).

The filter set contains the none filter for which $C_n^d = C_n$. The none filter turns RDLS into a regular lifting step if it is applied to all arguments of function f [see Eqs. (5) and (6)]. The heuristic we employ for an image-adaptive filter selection (see the next section) requires this filter to be present in the filter set.

We also used the null special filter case, for which $C_n^d = 0$. For the examined RDLS-modified color space transforms, the null filter may result in step skipping. If it is applied to all arguments of function f, then RDLS becomes $C_x \leftarrow C_x$ or $C_x \leftarrow -C_x$ and negating the image component does not change its entropy and has virtually no effect on its compression ratio.

It is noteworthy that the none and null filters may turn the RDLS-modified transform into a lifting transform or cause skipping it as a whole or partially—because different filters may be selected for different steps by the filter selection heuristic we employ. On the other hand, the specific RDLS may be only partially affected if its function f has more than one argument.

2.6 *Filter Selection Heuristic*

We used a simple and greedy denoising filter selection heuristic based on one applied in Ref. 11 for the RDLS-modified DWT transform. It consists of the below described steps: A and B.

A. Transform the image using the none filter in all steps for all arguments of function f. Store the estimated bitrates of the transformed components and, for each component in each step, assign the none filter.

B. In each transform step s (starting from step 1) for each function f argument C_n (analyzed in the $C_1, \ldots, C_3$ order), try to find a better denoising filter by checking for each filter (except for the one already selected), the overall estimated bitrate obtained by using in step s this filter for denoising of component C_n for all image pixels, while the filters selected so far are used for other components and in other steps.

Step B of the heuristic may be repeated for a given number of iterations. This step of the heuristic in each RDLS-modified forward transform step for each component requiring denoising greedily selects the denoising filter to be applied to all image pixels. For example, in step 3 of forward RDLS-RCT [Eq. (9)], first, a filter for C_1 is selected for denoising of all the image pixels and then a filter for C_3 is also selected for all pixels.

To obtain the estimated overall image bitrate after changing a filter, it may be sufficient to estimate bitrates of some components only—depending on the transform and the step, the changing of a denoising filter may not affect all components. Table 2 presents computational time complexities of the filter selection heuristic that take into account the above-mentioned property. Complexities are, for the investigated RDLS-modified transforms, expressed using the cost of operations on a single image component. For comparison, the complexities of compression using these transforms with the already selected filters and compression with the non-RDLS transforms are also reported. We assumed that all image pixels are used in bitrate estimation. Using for this purpose only some of the pixels allows lowering the complexity, which is discussed in the next section.

These estimations are rough, among others because the complexity of denoising may differ significantly for different filters and because different lifting steps are not equally complex (we took into account that step 3 of forward RDgDb is done at no cost and that no actual denoising is applied in step A of the heuristic). However, they allow making certain general observations. The complexity of the heuristic depends linearly on the number of iterations of step B, the number of filters, and on complexities of bitrate estimation, lifting, and denoising. It also depends on the RDLS-modified transform it selects filters for; it is the smallest for RDLS-RDgDb, whereas for others, it is about three (for RDLS-RCT and RDLS-LDgEb) or four (RDLS-YCoCg-R) times greater.

Table 2 Complexities of the heuristic and the compression exploiting transforms.

Transform	Heuristic	Compression (RDLS transform)	Compression (non-RDLS transform)
RDLS-RCT	$6h(f-1)(c_e+c_l+c_d)+3c_e+3c_l$	$3c_c+3c_l+4c_d$	$3c_c+3c_l$
RDLS-YCoCg-R	$8h(f-1)(c_e+c_l+c_d)+3c_e+3c_l$	$3c_c+3c_l+4c_d$	$3c_c+3c_l$
RDLS-RDgDb	$2h(f-1)(c_e+c_l+c_d)+3c_e+2c_l$	$3c_c+2c_l+2c_d$	$3c_c+2c_l$
RDLS-LDgEb	$6h(f-1)(c_e+c_l+c_d)+3c_e+3c_l$	$3c_c+3c_l+3c_d$	$3c_c+3c_l$

Note: c_d, cost of denoising of an image component; c_l, cost of modifying an image component by applying a lifting step to all image pixels; c_e, cost of estimating the bitrate for the component; c_c, cost of actual compression of a component; f, number of denoising filters; h, number of iterations of step B of the heuristic.

Assuming the perfect bitrate estimation, the step-by-step regime, and that for denoising of C_n of a specific pixel only C_n of this and of other pixels are used, the results obtained after a single iteration of step B of the heuristic are optimal in the case of the RDLS-RDgDb transform. In this transform, a component modified in a given step is modified based on only one other component and is not used in the next steps. Therefore, the filter selected for this step affects the bitrate of the component modified by it only.

The selected filters have to be passed to the decoder along with the compressed data. In this research, we initially used up to 13 filters (described in Sec. 2.5) and from two to four filters must be selected for an image depending on the applied transform. The cost of encoding the filter selection using a fixed-length binary code is at most 20 bits per image—it is negligible.

2.7 *Estimation of Component Compression Effects*

As the primary estimator of the image component compression effects, denoted H0_pMED, we used the memoryless entropy of the component residual image, i.e., of a single-component image consisting of prediction errors calculated as differences between actual and predicted component pixels. The bitrate of the three-component image was estimated as a sum of the estimated bitrates of its three components. The memoryless entropy of a single-component image (a residual image in this case) is $H_0=-\sum_{i=0}^{N-1}p_i\log_2 p_i$, where N is the alphabet size and p_i is the probability of occurrence of pixel value i in the image. For prediction, we used the nonlinear edge-detecting predictor MED [Eq. (13)],[9] which originates from the median adaptive prediction coding of video data:[13,14]

$$\mathrm{MED}(C_n^{[a,b]})=\begin{cases}\min(C_n^{[a-1,b]},C_n^{[a,b-1]}) & \text{if } C_n^{[a-1,b-1]}\geq\max(C_n^{[a-1,b]},C_n^{[a,b-1]})\\ \max(C_n^{[a-1,b]},C_n^{[a,b-1]}) & \text{if } C_n^{[a-1,b-1]}\leq\min(C_n^{[a-1,b]},C_n^{[a,b-1]})\\ C_n^{[a-1,b]}+C_n^{[a,b-1]}-C_n^{[a-1,b-1]} & \text{otherwise}\end{cases}, \tag{13}$$

where $C_n^{[a,b]}$ is the component C_n of the image pixel in column a and row b and MED $(C_n^{[a,b]})$ is its predicted value. For the top image row, we used $C_n^{a-1,b}$ as a predictor; for the leftmost column, we used $C_n^{a,b-1}$; and 0 was a predictor for the top left image pixel.

We also examined limited-complexity estimation methods, where for each transformed image component, instead of entropy of prediction errors of all the pixels, we used:

- the memoryless entropy of 10,000 pseudorandomly selected pixels' component prediction errors, this estimator is denoted H0_pMED(10k:1), and
- the memoryless entropy of 10,000 component prediction errors from 100 pseudorandomly selected nonoverlapping 10×10 pixels rectangles, denoted H0_pMED(10k:100).

For the selection of pixels in H0_pMED(10k:1) and H0_pMED(10k:100), we reinitialized the pseudorandom number generator each time the image component bitrate was estimated. Thus, from all components of an image, in all steps and iterations of the heuristic, the same pixels were used for estimation.

H0_pMED(10k:1) was found by Strutz to be a sufficient estimator for a close to the optimum color space transform selection.[5] For typical image sizes, compared to using for all image pixels, a simpler predictor or to computing the entropy of the image component instead of the component prediction error, it allows a greater reduction of the computational time complexity of the compression effects estimation for the lifting-based transforms.[15] Computing H0_pMED(10k:1) for the three-component image transformed with non-RDLS transform is of low complexity; in order to obtain 30,000 component prediction errors (10,000 in each component), we have to transform 40,000 pixels and compute the MED predictor 30,000 times, whereas the smallest image used in this study contained 262,144 pixels. In the case of the RDLS-modified transform, however, there appears the large extra cost of denoising operations necessary to obtain the prediction errors of individual pixels; denoising operations are the most important factor of the complexity of the heuristic employing H0_pMED(10k:1).

For example, 72 pixel components must be denoised in order to obtain the prediction errors of the single pixel components computed by RDLS-YCoCg-R, which was calculated as follows. We assumed that neighborhoods of

individual pixels whose prediction errors are selected for bitrate estimation are not overlapping or an accidental overlapping is not exploited to reduce the estimation cost and that we use denoising filters with 3×3 pixels windows. To obtain the MED prediction error of a transformed C_2 component of a pixel, which is computed in step 3 of RDLS-YCoCg-R [Eq. (10)], we need C_2 of this pixel and of its neighbors (upper, left-hand, and upper-left)—a 2×2 pixels rectangle area; these pixels are computed in step 3 based on a 2×2 pixels rectangle of C_3^d components (requiring four denoising operations). To obtain a 2×2 pixels rectangle of C_3^d components, we use a 4×4 pixels rectangle of transformed C_3 components that are computed in step 2. In step 2, to obtain a 4×4 pixels rectangle of C_3, we use 4×4 pixels rectangles of C_1^d and C_2^d (32 denoisings); C_2 in this step is an untransformed G primary color component, but C_1 is computed in step 1. To obtain a 4×4 pixels rectangle of C_1^d, we need a 6×6 pixels rectangle of transformed C_1, that, in step 1, is computed using a 6×6 pixels rectangle of C_3^d (36 denoisings). While computing the C_2 prediction error, we obtained data sufficient for computing prediction errors of the remaining components. All in all, we have to perform the pixel's component denoising 72 times per pixel and 720,000 times to get the H0_pMED(10k:1) estimation of the image compression ratio.

For RDLS-YCoCg-R and the smallest images used in this research, the number of denoising operations required by H0_pMED(10k:1) is not much smaller compared to H0_pMED, which requires four denoising operations per image pixel. To reduce the estimation cost, we proposed the H0_pMED(10k:100) estimator that requires 68,400 denoising operations to obtain the bitrate estimation of an RDLS-YCoCg-R transformed image (assuming the 3×3 pixels window of denoising filter and that the neighborhoods of the 10×10 pixels rectangles selected for bitrate estimation are not overlapping or that an accidental overlapping is not exploited to reduce the estimation cost).

As already noted, the heuristic does not require performing all of the transform steps after each filter change and not all components' bitrates must be estimated each time an overall image bitrate is estimated, which allows certain optimizations. Taking them into account, we present in Table 3 the cost of the heuristic for various RDLS-modified transforms, expressed as a number of denoisings of a component of a single pixel. The cost is calculated assuming that in step A of the heuristic, no actual denoising is used, but each time we estimate the bitrate in step B, we do it as if all filters used a 3×3 pixels window. In the cases when the heuristic cost for H0_pMED(10k:1) is the highest (i.e., for RDLS-YCoCg-R and RDLS-LDgEb), employing H0_pMED(10k:100) decreases it over eight times.

We also note that the filtering operation may be optimized. For example, computing the smoothing filter with a 3×3 pixels window and center point weight 1 for an individual pixel [which is needed for the H0_pMED(10k:1) estimator] requires nine arithmetic operations; for pixels inside a contiguous rectangular area [for H0_pMED and H0_pMED(10k:100)], the cost of this filter drops to five arithmetic operations. Having computed the smoothing filter with a certain center point weight, computing it for some other weight requires just three arithmetic operations.

2.8 *Test Data, Implementations, and Procedure*

The evaluation was performed for the sets of 8-bit RGB test images listed below.

- Waterloo—a set of eight color images from the University of Waterloo, image sizes range from 512×512 to 1118×1105 pixels;[16]
- Kodak—a set of 23 images released by the Kodak corporation, all images are of size 768×512 pixels;[17]
- EPFL—a set of 10 images used at the École polytechnique fédérale de Lausanne for subjective JPEG XR quality evaluation,[18] images sizes: 1280×1506 to 1280×1600 pixels;[19]
- A1—a set of three large images scanned from a 36-mm high quality diapositive film, image sizes range from 7376×4832 to 7424×4864 pixels;[20]
- A2—a set of 17 images acquired from 36 mm negatives, image sizes are from 1620×1128 to 1740×1164 pixels;[20]
- A3—a set of 116 images acquired using a camera equipped with a Bayer-pattern RGGB color filter array, all images are of size 1992×1493 pixels;[20]
- A1-red.3, A2-red.3, and A3-red.3—sets of reduced size (3×) images from sets A1, A2, and A3, respectively.

The sets A1, A2, and A3 contain unprocessed photographic images in optical resolutions of acquisition devices, or (A3) as close to such resolution as possible without

Table 3 The cost of the heuristic for various bitrate estimation methods and the cost of the actual RDLS transforms, expressed in number of denoising operations of a pixel's component. The heuristic complexity is reported for denoising filters using up to 3×3 pixels windows.

Transform	Heuristic cost			Transform cost
	H0_pMED	H0_pMED(10k:1)	H0_pMED(10k:100)	
RDLS-RCT	$6t(f-1)h$	$480000(f-1)h$	$82200(f-1)h$	$4\,t$
RDLS-YCoCg-R	$8t(f-1)h$	$1000000(f-1)h$	$121600(f-1)h$	$4\,t$
RDLS-RDgDb	$2t(f-1)h$	$80000(f-1)h$	$24200(f-1)h$	$2\,t$
RDLS-LDgEb	$6t(f-1)h$	$800000(f-1)h$	$92600(f-1)h$	$3\,t$

Note: t, number of pixels in the image; f, number of denoising filters; h, number of iterations of step B of the heuristic.

interpolation of all components. Except for Waterloo, all images may be characterized as continuous-tone photographic. The most widely known Waterloo set contains both photographic and artificial images; some of them are dithered, sharpened, computer-generated, composed of others, or have globally or locally highly sparse histograms of intensity levels.[21,22] The same image sets were used for experiments in Ref. 3, where their more detailed characteristics may be found.

RDLS effects on bitrates were analyzed for three significantly different standard image compression algorithms in the lossless mode: the predictive JPEG-LS,[6,9] the DWT-based JPEG 2000,[7,23] and the JPEG XR employing the discrete cosine transform.[8,24,25] We used the Signal Processing and Multimedia Group, Univ. of British Columbia JPEG-LS implementation, version 2.2,[26] the JasPer implementation of JPEG 2000 by M. Adams, version 1.900,[27,28] and the JPEG XR reference software.[29]

All algorithms were used to compress individual transformed components, one component at a time. Due to requirements of employed file formats and implementations, all components were stored using non-negative integers. Components transformed with the lifting transforms were stored using the nominal component bit-depths. See Table 1 for nominal depths of components of all the examined, lifting-based and RDLS-modified transforms. Since in initial tests, the greater nominal depth of the RDLS-modified transform was rarely needed, for these transforms, we used the bit-depth of the lifting counterpart or, only if pixels of an actual transformed image component exceeded this depth, we increased the component bit-depth up to the nominal depth of the RDLS-modified transform component. The implementation used for applying transforms and the heuristic is freely available.[30] The compression ratio or bitrate r, expressed in bits per pixel (bpp), is calculated based on the total size in bytes of the individually and independently compressed three components of the transformed image, including compressed file format headers; smaller r denotes better compression.

3 Results and Discussion

3.1 *Reversible Denoising and Lifting Steps Effects on Color Space Transforms*

In Fig. 3, we present the RDLS effects on the bitrates of individual RDLS-RCT transformed components and on the overall image bitrates of each of the examined transforms. Bitrates for non-RDLS and RDLS-modified transforms, obtained using denoising filters selected based on H0_pMED estimator in three iterations of step B of the heuristic, were averaged for each set. For easier comparison of RDLS effects, in figures we show the bitrate changes due to RDLS expressed as the percentages of the bitrates obtained with a non-RDLS transform, whereas the absolute bitrates of selected variants of transforms are presented in tables. We also report an average for all sets, however, calculated using average of set-averages, not the direct average of all images. The A3 and A3-red.3x sets contain many more images (116 in each) than all other sets (81 images); therefore, a simple average would be biased toward the A3 and A3-red.3x.

Figures 3(a)–3(c) show that for RDLS-RCT, the overall bitrate improvement due to RDLS is, in many cases, a result of improved bitrates of chrominance components and worsened bitrates of luminance. The above is also true for RDLS-YCoCg-R and RDLS-LDgEb (not shown in Fig. 3). The overall three-component improvements for the RDLS-RCT transform [Fig. 3(d)] result from both employing the actual denoising filters [Fig. 3(e)] and the step skipping by applying the null filter to RDLS [Fig. 3(f)]. Looking at the RDLS effects for other transforms [Fig. 3(g)–3(i)], we see that the greatest bitrate improvements of over 2% on average for all sets were obtained for RDLS-RDgDb. For this transform, the improvements for chrominance components are not accompanied by a worsened bitrate of the third component (i.e., the unmodified primary color R). Generally, the bitrate improvements due to application of RDLS to a color space transform may significantly differ for different sets and for different transforms in the case of a specific set, but are similar for different compression algorithms. Similarity among compression algorithms is stronger when we do not use the null filter. RDLS effects are less pronounced for the JPEG XR algorithm, especially in the case of Waterloo images.

We examined the RDLS effects on color space transforms using several compression algorithms and test image sets. For brevity, we focus on results of the most popular JPEG 2000 algorithm, averaged for all sets. In Table 4, for various transform variants, we report both the bitrates obtained by the RDLS-modified transforms and the bitrate improvements with respect to the non-RDLS transform. To provide a single measure allowing comparisons of variants, we also report the RDLS bitrate change averaged for all the transforms (column labeled "All"). Employing only the transform step skipping, implemented as a special case of the RDLS [i.e., using a null filter and not using smoothing filters, row "RDLS (no smoothing)"], allows bitrate improvements comparable to those obtained by RDLS with the typical denoising and without the step skipping [row "RDLS (no null)"]—better for RCT and YCoCg-R, worse for RDgDb and LDgEb. Step skipping results are better than results obtained by simply deciding, based on the estimated bitrate, whether to perform unmodified transform or to skip it [row "min(RGB, non-RDLS)"], although for YCoCg-R, the simpler method is better. Finally, employing both the step skipping and typical denoising filters allows significantly larger RDLS bitrate improvements, than those obtained when using only the typical denoising filters (compare the two last rows in Table 4). Compared to the filter selection heuristic from Ref. 10 (row "RDLS (Ref. 10, no null)"), for the same filter set, the heuristic we propose in this study results in greater bitrate improvements [row "RDLS (no null)"] and it is of significantly lower complexity in the case of RCT and YCoCg-R.

The average level of improvement of over 1% that we obtained for RCT, YCoCg-R, and LDgEb by using all the denoising filters described in Sec. 2.5 selected based on the H0_pMED estimator in three iterations of the heuristic step B is not negligible as for lossless image compression. However, the heuristic cost may be too high for practical applications. Significantly larger improvements were obtained for specific sets and in the case of RDgDb. In the next section, we reduce the heuristic cost without sacrificing most of the bitrate improvements.

For the above variant, we checked the actual bit-depth expansion of the transformed components. For each

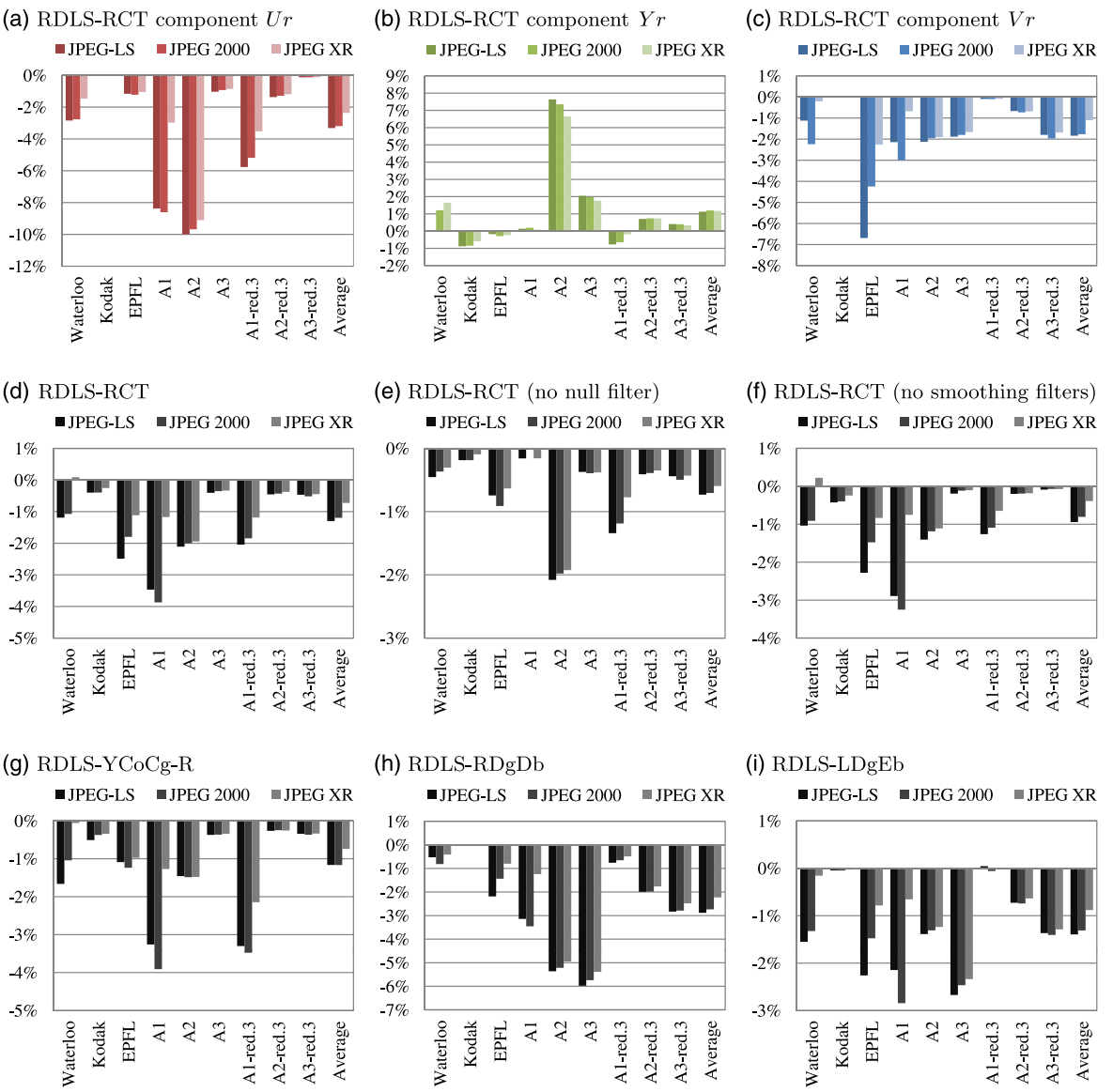

Fig. 3 (a–c) Average JPEG 2000 bitrate changes due to RDLS, for the individual RDLS-RCT components, and (d–i) the overall for examined transforms, obtained using denoising filters described in Sec. 2.5 (all, unless indicated otherwise) selected in three iterations of step B of the heuristic based on bitrates estimated with H0_pMED.

Table 4 Comparison of the RDLS and non-RDLS transform variants. Unless indicated otherwise, denoising filters were selected in three iterations of heuristic step B from all filters described in Sec. 2.5; H0_pMED was employed for the RDLS filter selection and in min(RGB, non-RDLS) non-RDLS variant for choosing between performing unmodified transform or skipping it; RDLS (Ref. 10, no null), filters selected as in Ref. 10.

	RCT		YCoCg-R		RDgDb		LDgEb		All	
Transform variant	r	Δr	r	Δr	r	Δr	r	Δr	r	Δr
non-RDLS transform	11.4374	0.00%	11.4977	0.00%	11.6173	0.00%	11.5039	0.00%	11.5141	0.00%
min(RGB, non-RDLS)	11.3890	−0.42%	11.3921	−0.92%	11.4460	−1.47%	11.4637	−0.35%	11.4227	−0.79%
RDLS (no smoothing)	11.3452	−0.81%	11.4174	−0.70%	11.4067	−1.81%	11.4113	−0.81%	11.3951	−1.03%
RDLS (Ref. 10, no null)	11.3704	−0.59%	11.5184	0.18%	11.3333	−2.44%	11.4640	−0.35%	11.4215	−0.80%
RDLS (no null)	11.3568	−0.71%	11.4345	−0.55%	11.3333	−2.44%	11.4000	−0.90%	11.3812	−1.15%
RDLS	11.3004	−1.20%	11.3635	−1.17%	11.2995	−2.74%	11.3531	−1.31%	11.3291	−1.61%

Note: r—JPEG 2000 bitrate, average for all sets (bpp); Δr—bitrate change with respect to the non-RDLS transform.

transform that may result in a component bit depth greater than the non-RDLS counterpart (RDLS-RCT, RDLS-YCoCg-R, and RDLS-LDgEb), such expansion happened for about two thirds of the images in the case of the luminance component, which each time was expanded by 1 bit.

3.2 *Reducing the Complexity of the Filter Selection*

As shown in Sec. 2.6, the heuristic complexity is proportional to the number of iterations of its step B and to the number of denoising filters. Therefore, we investigated decreasing the iterations number and smaller filter sets. For RDLS-RDgDb, a single iteration of heuristic step B results in the optimal filter selection; bitrate improvements due to RDLS obtained for other transforms in 1, 2, and 3 iterations are presented in Fig. 4. In practice, two iterations seem sufficient; the average bitrates for all sets, obtained in two and three iterations, do not differ noticeably and for individual sets only in three cases the bitrate differs by more than 0.1 percentage points (for the A1.red.3 set in the case of RDLS-RCT and RDLS-YCoCg-R and for A1 in the case of the former transform). On the other hand, by using only one iteration, as compared to two iterations, average bitrates for all sets get over 0.1 percentage point worse for RDLS-YCoCg-R and RDLS-LDgEb, whereas for individual sets, one iteration may be worse by over 1 percentage point. Thus, remembering that in the case of RDLS-RDgDb, the single iteration is optimal, we use two iterations as a starting point for testing other options of the complexity reduction.

In Table 5, we report the JPEG 2000 bitrate changes for a couple of reduced complexity filter selection variants. As previously, we also report the RDLS bitrate change averaged for all the transforms in the column labeled "All." Among others, we examined reducing the number of denoising filters by rejecting of some of the smoothing filters. By using filters with center point weights being even powers of 2 in range from 1 to 256 (row labeled "2 iterations, 7 filters, H0_pMED"), instead of integer powers in range from 1 to 1024, we decrease the complexity of the heuristic about two times (as the filter number drops from 13 to 7) at the acceptable cost of a smaller compression ratio improvement by below 0.05 percentage points on average for all transforms. Further reducing the set by using only three smoothing filters (row "2 iterations, 5 filters, H0_pMED") results in a smaller complexity decrease and a greater cost. Therefore, for the former variant, we applied the simplified compression effect estimation methods.

The differences between effects of H0_pMED and H0_pMED(10k:1) estimators are negligible. On average for all transforms, they are below 0.005 percentage points. Interestingly, the case of the RDLS-LDgEb transform the H0_pMED(10k:1) estimator results in a better average bitrate for all sets than H0_pMED (better by 0.02 percentage

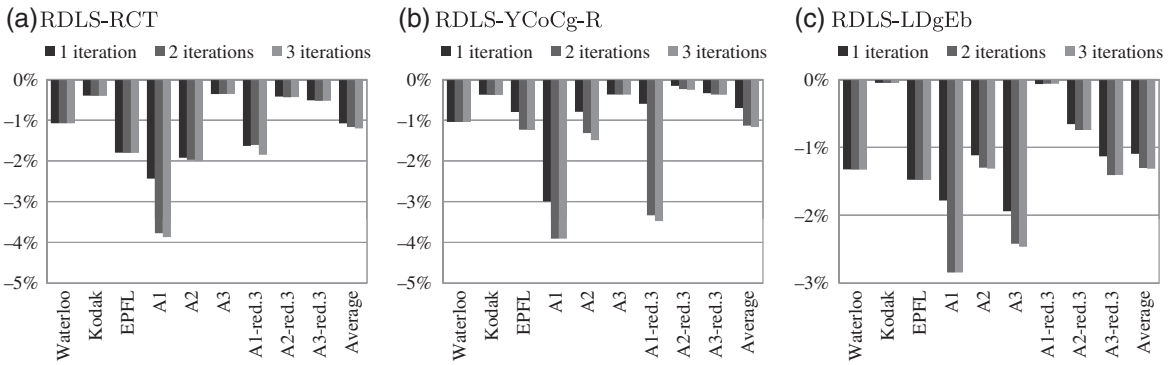

Fig. 4 Average JPEG 2000 bitrate changes due to RDLS obtained using denoising filters selected in 1, 2, and 3 iterations of the heuristic step B out of all filters described in Sec. 2.5. (a) RDLS-RCT, (b) RDLS-YCoCg-R, and (c) RDLS-LDgEb.

Table 5 Reduced complexity RDLS filter selection variants. The JPEG 2000 bitrate changes with respect to non-RDLS transforms are reported. Denoising filters were selected from all or some of the filters described in Sec. 2.5 (13—all; 7—none, null, and smoothing with center point weights 1, 4, 16, 64, and 256; 5—none, null, and smoothing with center point weights 1, 16, and 256).

Filter selection variant	RDLS-RCT	RDLS-YCoCg-R	RDLS-RDgDb	RDLS-LDgEb	All
3 iterations, 13 filters, H0_pMED	−1.20%	−1.17%	−2.74%	−1.31%	−1.61%
2 iterations, 13 filters, H0_pMED	−1.17%	−1.13%	−2.74%	−1.30%	−1.59%
2 iterations, 7 filters, H0_pMED	−1.13%	−1.11%	−2.68%	−1.27%	−1.55%
2 iterations, 5 filters, H0_pMED	−1.08%	−0.98%	−2.57%	−1.19%	−1.46%
2 iterations, 7 filters, H0_pMED(10k:1)	−1.13%	−1.10%	−2.68%	−1.29%	−1.55%
2 iterations, 7 filters, H0_pMED(10k:100)	−1.02%	−1.09%	−2.65%	−1.26%	−1.51%

points). Our results show that the close to the optimum performance of H0_pMED(10k:1), first observed by Strutz for lifting color space transforms, is a more general property of this estimator. Note that by the optimum performance, we mean estimation effects obtained using H0_pMED; in the next section, we check, among others, how good H0_pMED estimation is compared to the actual bitrate of the actual compression algorithm.

The H0_pMED(10k:100) estimator we proposed in order to lower the bitrate estimation cost results in bitrates little worse than H0_pMED(10k:1); on average, for all transforms, it is by 0.04 percentage points worse, for the RDLS-RCT (the worst case) by 0.11 percentage points. The H0_pMED(10k:100) appears to be the most interesting general purpose estimator from a practical standpoint—it allows fast heuristic filter selection that results in bitrates that are close to the bitrates obtained using the most complex variant examined so far (using H0_pMED, three iterations, and all filters described in Sec. 2.5). Compared to the latter, we get bitrates worse by 0.1 percentage points on average for all transforms and sets in the case of the JPEG 2000 coding.

We do not report the actual filter search time of the heuristic or the time of transforming the image with RDLS-modified transforms because our research implementation was not optimized; among others, we did not exploit the possibility of partial estimation of the image bitrate after changing a single filter by the heuristic and each time before outputting a transformed image, the transform reversibility was verified by performing an inverse transform. However, knowing the parameters of the heuristic, its cost may be compared to the cost of the transform it selects filter for. For the smallest images we used (containing 262,144 pixels), with respect to the number of denoising operations (that is to the most expensive part of the heuristic). In Table 6, we report the transform cost and the heuristic cost. The heuristic cost is reported for the H0_pMED(10k:100)-based selection of filters done in two iterations of step B from the set of seven filters using up to 3×3 pixels windows. The heuristic cost is generally close to the transform cost. It is by 6% lower in the case of RDLS-RCT, for RDLS-YCoCg-R and RDLS-LDgEb, it is by about 40% higher, and for RDLS-RDgDb, it is two times lower. Note that for the latter transform, we may use only one iteration as it will not affect the filter selection and further decrease the cost two times. For this transform also, the H0_pMED(10k:1) estimator is practically acceptable, as for one iteration of step B, it requires 480,000 denoisings, i.e., 92% of the number of denoisings required by the RDLS-RDgDb transform. For larger images, the cost of the heuristic exploiting H0_pMED(10k:100) or H0_pMED(10k:1) remains constant, whereas the transform cost grows in direct proportion to the number of pixels in the image.

Table 6 The cost of the heuristic for the H0_pMED(10k:100) estimator and the cost of the actual RDLS transforms, calculated for the smallest images (262144 pixels) and expressed in number of pixel component denoisings. Denoising filters were selected in 2 iterations of the heuristic step B out of the set of 7 filters using up to 3×3 pixels windows.

Transform	Heuristic cost	Transform cost
RDLS-RCT	986,400	1,048,576
RDLS-YCoCg-R	1,459,200	1,048,576
RDLS-RDgDb	288,240	524,288
RDLS-LDgEb	1,111,200	786,432

To verify how the modifications we selected based on the RDLS effects on JPEG 2000 coding affect other algorithms, in Table 7, we report bitrates and bitrate changes obtained for different compression algorithms. Simplifying the estimation method and reducing the size of the filter set and the number of iterations did not change the general way RDLS affects bitrates in the case of different algorithms and color space transforms. RDLS effects for JPEG-LS and JPEG 2000 are close to each other, whereas for the JPEG XR algorithm, the improvements are smaller. The bitrate improvement due to RDLS is the greatest in the case of RDgDb (2.15% for JPEG XR, 2.65% or more for JPEG-LS and JPEG 2000), whereas the bitrates of more complex transforms were improved by over 1% in the case of JPEG-LS and JPEG 2000, and by over 0.5% for JPEG XR. In Fig. 5, the bitrate changes for individual sets are presented and compared to a variant employing H0_pMED estimation, three iterations and a larger set of denoising filters. The greatest differences in effects for these variants may be noticed for RDLS-RCT and RDLS-YCoCg-R in the case of some sets only. In a single case of the former transform for Waterloo images and the JPEG XR algorithm, the simplified filter selection results in about 1.2% worse bitrates than the non-RDLS transform; for this algorithm, images, and transform, the more complex filter selection variant resulted in bitrate worsening (by below 0.1%). For RDLS-RDgDb and RDLS-LDgEb, the results of the simplified filter selection

Table 7 Effects of the H0_pMED(10k:100)-based filter selection variant on JPEG-LS, JPEG 2000, and JPEG XR bitrates. Reported are: the bitrates for the RDLS-modified transforms (r) and the bitrate changes with respect to the non-RDLS transform (Δr), average for all sets and all sets and transforms (All). The denoising filters were selected in two iterations of the heuristic step B out of: none, null, and smoothing filters with center point weights 1, 4, 16, 64, and 256.

	RDLS-RCT		RDLS-YCoCg-R		RDLS-RDgDb		RDLS-LDgEb		All	
Algorithm	r	Δr	r	Δr	r	Δr	r	Δr	r	Δr
JPEG-LS	10.8840	−1.13%	10.9616	−1.10%	10.8621	−2.82%	10.9075	−1.35%	10.9038	−1.60%
JPEG 2000	11.3208	−1.02%	11.3726	−1.09%	11.3089	−2.65%	11.3594	−1.26%	11.3404	−1.51%
JPEG XR	12.4776	−0.51%	12.5078	−0.66%	12.4494	−2.15%	12.5056	−0.83%	12.4851	−1.04%

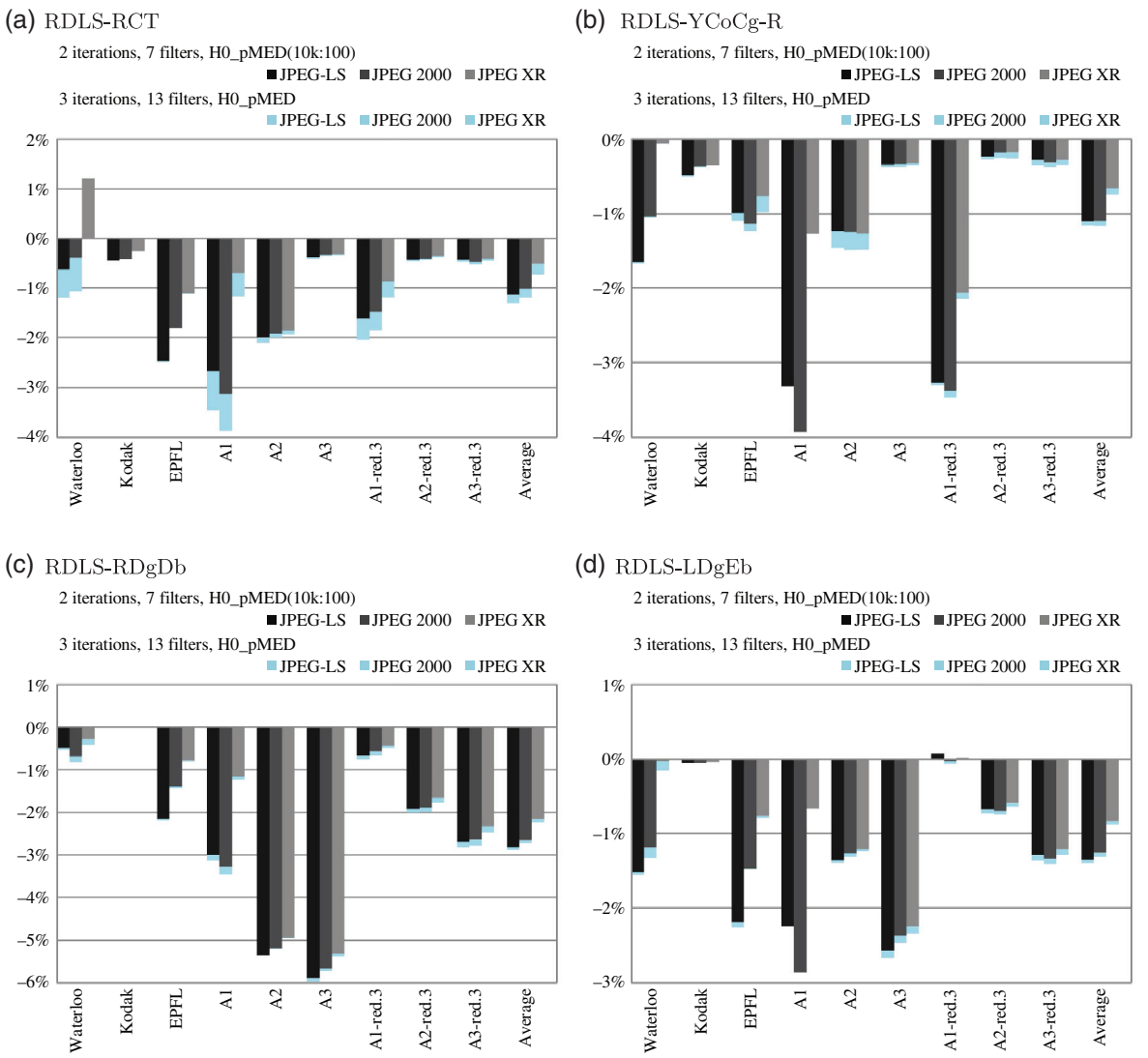

Fig. 5 Average JPEG-LS, JPEG 2000, and JPEG XR bitrate changes due to RDLS with respect to the non-RDLS transform. The denoising filters were selected using the H0_pMED(10k:100) estimator in two iterations of the heuristic step B out of: none, null, and smoothing filters with center point weights 1, 4, 16, 64, and 256, or using H0_pMED in three iterations of the heuristic step B out of all filters described in Sec. 2.5. (a) RDLS-RCT, (b) RDLS-YCoCg-R, (c) RDLS-RDgDb, and (d) RDLS-LDgEb.

for individual sets are similarly close to effects of the more complex variant as on average for all sets.

Looking at the absolute bitrates (Table 7), we notice that differences between compression algorithms for a specific transform are much larger than differences between transforms for a given algorithm. JPEG-LS is consistently the best, JPEG XR the worst. RDLS-RDgDb obtains the best average bitrates for each algorithm which is an effect of the greatest bitrate improvement due to RDLS. In the case of unmodified transforms (recall Table 4), on average for all sets we used in this research, RCT was the best. The bitrate improvements that we attained at a reduced cost appear worthwhile from a practical standpoint. For example, for the lossless JPEG 2000, the bitrates were improved on average for all sets and all transforms by about 1.5%. This improvement is not small if we consider that the best unmodified transform we evaluated (RCT) obtained an average bitrate better than the worst one (RDgDb) by about 1.2% only.

3.3 *Additional Experiments*

We started experiments using the H0_pMED estimator that was found effective for a non-RDLS color space transform selection[5,15] as well as for the selection of denoising filters for some of the RDLS-modified color space transforms investigated in this study.[3,10] We found that by using H0_pMED(10k:100), the estimation complexity may be significantly reduced without sacrificing the RDLS bitrate improvement. But how far from perfect is the H0_pMED or H0_pMED(10k:100) estimation in the case of the investigated RDLS-modified transforms? To check it, we compared the estimation effects to using in the filter selection heuristic the actual image compression algorithm instead of estimating its results (see top three rows in Table 8). The estimation effects are very good from a practical standpoint. Using the actual compressor results in bitrate improvements better than estimation-based by less than 0.1 percentage point on average for all transforms; the greatest difference is for RDLS-RCT, where the H0_pMED(10k:100)-based

Table 8 RDLS effects for additional filter selection variants. Reported are the average JPEG 2000 bitrate improvements with respect to the non-RDLS transform. The filter set contained the following seven filters: none, null, and smoothing with center point weights 1, 4, 16, 64, and 256. exhaustive, using exhaustive filter search instead of the heuristic; *r*_JPEG_2000, using for filter selection the actual JPEG 2000 bitrate instead of the estimated one.

Filter selection variant	RDLS-RCT	RDLS-YCoCg-R	RDLS-RDgDb	RDLS-LDgEb	All
2 iterations, 7 filters, H0_pMED(10k:100)	−1.02%	−1.09%	−2.65%	−1.26%	−1.51%
2 iterations, 7 filters, H0_pMED	−1.13%	−1.11%	−2.68%	−1.27%	−1.55%
2 iterations, 7 filters, *r*_JPEG_2000	−1.22%	−1.06%	−2.72%	−1.35%	−1.59%
exhaustive, 7 filters, *r*_JPEG_2000	−1.32%	−1.48%	−2.72%	−1.39%	−1.73%

bitrate is by 0.2 percentage point worse. Interestingly, for RDLS-YCoCg-R, the use of actual compressor results in bitrates that are worse than estimation-based. Results of the heuristic may not be optimal even when we use the perfect estimation.

The heuristic finds optimal filters for the RDLS-RDgDb transform and for this transform, we obtained the greatest bitrate improvement with respect to the non-RDLS transform. Should the RDLS effects on RDgDb be attributed to the imperfect heuristic filter selection in the case of more complex transforms, or is RDLS the most effective for the simplest transform also when for all transforms we employ optimal filters? We performed the exhaustive filter search and selected for each image and transform the optimal filters out of the set of seven denoising filters. For seven filters, such a search is impractical but realizable, as for the most complex transforms, RDLS-RCT and RDLS-YCoCg-R, it involves testing 2401 filter combinations per image. In Table 8 (row labeled "exhaustive,...") we report the effects of the RDLS-modified transforms for the optimal filter selection based on the actual JPEG 2000 compression bitrate. Let us compare the effects of the heuristic when employing two iterations of step B and actual compression instead of bitrate estimation, to optimal filter selection. The effects of the heuristic significantly vary for different transforms. Indeed, for RDLS-RDgDb, the heuristic filter selection is optimal. For the RDLS-RCT and RDLS-LDgEb, the heuristic is by 0.1 and 0.04 percentage points, respectively, worse than the optimum. However, for RDLS-YCoCg-R, the heuristic result is worse by about 1/4 than the bitrate improvement obtainable for this transform with optimal filter selection. A heuristic based on a different filter search strategy might be better in the case of RDLS-YCoCg-R. Also in the case of the optimal filter selection for all transforms, the largest improvement due to RDLS is for RDgDb—also for the optimal filter selection, RDLS is the most effective for RDgDb.

RDLS recently was found effective for a transform that is much more complex then a color space transform and involves more interdependent steps, i.e., for the multilevel 2-D DWT transform in lossless JPEG 2000 compression.[11] The bitrate improvements exceeding 13% were observed for grayscale images of nonphotographic content when the nonlinear denoising filters were applied. We checked if those filters could be used to further improve the effects of the RDLS-modified color space transforms. We tested all the additional filters from Ref. 11 that were not already used in this study:

- Smoothing filters, with 5×5 pixels windows, employing the same weights of the window center point, as before (11 filters).
- Median—two median filters (3×3 and 5×5 pixels windows), the median 5×5 pixels filter was the strongest (the most harsh) filter used in Ref. 11 and it was found the most effective in the bitrate improvement.
- RCRS-1—two filters (3×3 and 5×5 pixels windows), which belong to a general family of rank-conditioned rank selection (RCRS) filters.[31] RCRS-1 filters replace a sample with the window median if the sample is greater than or smaller than all other samples in the window.
- RCRS-2—two filters (3×3 and 5×5 pixels windows) that replace a sample with the second greatest window sample value if the sample is greater than the median and the greatest; or, if it is smaller than the median and the smallest, they replace a sample with the second smallest window sample value.

Since only the Waterloo set contains nonphotographic images, in Table 9, we report the effects of extending the filter set for RDLS-modified color space transforms with the above filters for both the Waterloo set and average for all the sets. Unfortunately, such a naive approach did not result in practically useful bitrate improvements, especially if we consider the increased complexity of selecting the filters from the set of 30 denoising filters containing filters with larger windows.

RDLS improvements differ for various sets, thus it could be expected that by finding filters better matching the actual image characteristics, greater bitrate improvements due to RDLS could be obtained. Instead of basing on an estimated component bitrate, the noise parameters might be estimated and the denoising filters might be selected based directly on the analysis of the component to be denoised. For a specific acquisition device, the device model may be constructed that allows determining the denoising filters based directly on the acquisition process parameters (e.g., see Refs. 32 and 33). The former approach has an additional advantage. The component that is available as the function f argument for filters selection in RDLS in forward transform [Eq. (6)], is also available for inverse RDLS in inverse transform [Eq. (7)]. Signaling the filter selection (or parameters of a more sophisticated filter) to the decoder might be avoided at the cost of increased decoder complexity, as the same filters, or filter

Table 9 RDLS effects for a larger set of denoising filters. The JPEG 2000 bitrate improvements with respect to the non-RDLS transform are reported, average for all sets and for the Waterloo set. All the filters described in Sec. 2.5 (13) or these filters and the additional filters described in Sec. 3.3 (total 30) were selected using H0_pMED in three iterations of the heuristic step B.

Filters	Sets	RDLS-RCT	RDLS-YCoCg-R	RDLS-RDgDb	RDLS-LDgEb	All
13	All sets	−1.20%	−1.17%	−2.74%	−1.31%	−1.61%
30	All sets	−1.22%	−1.23%	−2.78%	−1.34%	−1.65%
13	Waterloo only	−1.07%	−1.04%	−0.81%	−1.33%	−1.06%
30	Waterloo only	−1.08%	−1.03%	−0.83%	−1.34%	−1.07%

parameters, can be found by the decoder based on analysis of the same data. On the other hand, in Ref. 10, we used a heuristic that performed an exhaustive search of filters in a given step based on the estimated bitrate of the component modified by this step only and in the case of RDLS-RCT and RDLS-LDgEb, it resulted in significant worsening of bitrates of some images. Therefore, it may be expected that selecting a filter for the data to be denoised based on this data may be effective for the simplest RDLS-RDgDb transform and not necessarily for others, which is an interesting topic that we leave for future research.

4 Summary

In this study, we examined the application of RDLS to the RCT, YCoCg-R, RDgDb, and LDgEb color space transforms; the RDLS-modified transforms are named RDLS-RCT, RDLS-YCoCg-R, RDLS-RDgDb, and RDLS-LDgEb, respectively. For the image-adaptive denoising filter selection, we proposed a simple and greedy heuristic consisting of steps A and B, where step B may be performed for a given number of iterations. In the heuristic, we used compression algorithm independent estimators of the filter selection effects on the transformed image bitrate. Initially, we used an estimator based on the memoryless entropy of the transformed image MED prediction errors of all pixels of each component (H0_pMED). We also employed a simplified, limited computational time complexity estimator that uses 10,000 pseudorandomly selected pixels [H0_pMED(10k:1)]. To further decrease the complexity of bitrate estimation, we proposed the H0_pMED(10k:100) estimator that also uses 10,000 pixels, but due to grouping them, in the case of the most complex RDLS-modified color space transforms, its complexity is over eight times lower compared to the H0_pMED(10k:1). Beside typical denoising filters (11 linear smoothing filters with 3×3 pixels windows) and the none filter, that may turn RDLS into the regular lifting step, we proposed the special filter case named the null filter. For the null filter, the denoised sample equals 0, which may result in the step skipping. In the experiments, we used several test image sets and significantly different standard image compression algorithms in the lossless mode: JPEG-LS, JPEG 2000, and JPEG XR.

We found that generally, the RDLS effects significantly differ for different image sets and for different transforms in the case of a specific set. They are similar for different compression algorithms, but they are less pronounced in the case of the JPEG XR algorithm. The largest average bitrate improvements were obtained for the simplest transform RDLS-RDgDb and improvements for others were roughly two times smaller. The overall bitrate improvements due to RDLS result from employing of both the actual denoising filters and the null filter. Although a certain level of bitrate improvement might be obtained by simply checking the estimated effects of skipping the entire color space transform, greater improvements were obtained by employing the null filter that may result in a partial transform skipping. The initial number of smoothing filters could be reduced without sacrificing the bitrate improvements. Assuming the perfect bitrate estimation, due to properties of RDLS-RDgDb, the proposed heuristic in one iteration of step B finds optimal filters for this transform. For other transforms, performing two iterations of step B is justified as using more iterations does not improve RDLS effects noticeably.

The most expensive element of the computational time complexity of the heuristic is the denoising of pixel components. This cost may be limited and reduced by employing simplified compression effect estimators. When using the H0_pMED(10k:100) estimator, the heuristic cost (for two iterations of its step B and seven denoising filters) gets close to the transform cost for RDLS-RCT, RDLS-YCoCg-R, and RDLS-LDgEb transforms. The heuristic cost for RDLS-RDgDb is four times lower than the transform cost (here, one iteration suffices) and for this transform, the cost of the more expensive H0_pMED(10k:1) estimator is still lower than the transform cost. For larger images, the cost of the heuristic exploiting H0_pMED(10k:100) or H0_pMED(10k:1) remains constant, whereas the transform cost (and the cost of the heuristic exploiting H0_pMED) grows in direct proportion to the number of pixels in the image. The H0_pMED(10k:100)-based heuristic filter selection results in at least about three fourths of the bitrate improvement obtainable with RDLS for the optimal filter selection based on the actual bitrate of the compression algorithm.

All in all, the most interesting results from a practical standpoint were obtained for an image-adaptive heuristic filter selection from the set of seven filters (none, null, and smoothing with center point weights 1, 4, 16, 64, and 256) using the simplified estimator of compression effects H0_pMED(10k:100), which is independent of the actually employed compression algorithm. On average, the bitrate improvement due to RDLS is the greatest in the case of RDLS-RDgDb (2.65% or more for JPEG-LS and JPEG 2000, 2.15% for JPEG XR), while the bitrates of more complex transforms were improved by over 1% in the case of JPEG-LS and JPEG 2000, and by over 0.5% for JPEG XR. For some sets, the improvements due to RDLS-RDgDb exceed 5%. Also, with respect to the absolute bitrate, this transform was the best for all the image compression

algorithms investigated in this study. In addition to the better average bitrates and bitrate improvements as well as the lower filter selection cost, another advantage of this transform is that its dynamic range is not increased compared to the non-RDLS counterpart.

We suppose that by finding filters better matching the actual image characteristics greater bitrate improvements due to RDLS could be obtained. For a given acquisition device, the denoising filters may probably be selected based on the acquisition process parameters rather than by using the heuristic employing estimated compression ratio of the denoised component. For RDLS-RDgDb, they may be selected or constructed directly based on the component being denoised, thus avoiding the need to signal to the decoder the filter selection.

5 Conclusion

RDLS applied to color space transforms allows the improvement of the bitrates of lossless image compression algorithms, however, RDLS effects depend on selecting proper denoising filters for the image being compressed. By exploiting the new contributions of this study, i.e., the filter-selection heuristic and the special filter case (the null filter), we attained bitrate improvements that on average are about two times higher than those obtained using the previously reported methods. Another new contribution, the H0_pMED(10k:100) compression effect estimator, reduced the cost of the filter selection process without sacrificing the majority of the bitrate improvement. The filter selection cost gets this way close to or lower than the transform cost, while on average for all the investigated transforms and images, the lossless JPEG 2000 bitrates are improved by about 1.5%; bitrates of certain images are improved to a significantly greater extent. All in all, the RDLS-modified color space transforms appear worthwhile from a practical standpoint.

Acknowledgments

This work was supported by the BK-219/RAU2/2016 grant from the Institute of Informatics, Silesian University of Technology and by the 02/020/RGH15/0060 grant from the Silesian University of Technology. A patent application on the RDLS-modified color space transforms presented in this work was filed to the Polish Patent Office (application number: P.396766).

References

1. H. S. Malvar, G. J. Sullivan, and S. Srinivasan, "Lifting-based reversible color transformations for image compression," *Proc. SPIE* **7073**, 707307 (2008).
2. W. Sweldens, "The lifting scheme: a custom-design construction of biorthogonal wavelets," *Appl. Comput. Harmonic Anal.* **3**, 186–200 (1996).
3. R. Starosolski, "Reversible denoising and lifting based color component transformation for lossless image compression," arXiv:1508.06106 [cs.MM], http://arxiv.org/abs/1508.06106 (2016) (Submitted).
4. R. Starosolski, "New simple and efficient color space transformations for lossless image compression," *J. Vis. Commun. Image Represent.* **25**(5), 1056–1063 (2014).
5. T. Strutz, "Multiplierless reversible colour transforms and their automatic selection for image data compression," *IEEE Trans. Circ. Syst. Video Technol.* **23**(7), 1249–1259 (2013).
6. ISO/IEC and ITU-T, "Information technology–Lossless and near-lossless compression of continuous-tone still images–Baseline," ISO/IEC International Standard 14495-1 and ITU-T Recommendation T.87 (2006).
7. ISO/IEC and ITU-T, "Information technology–JPEG 2000 image coding system: core coding system," ISO/IEC International Standard 15444-1 and ITU-T Recommendation T.800 (2004).
8. ISO/IEC and ITU-T, "Information technology–JPEG XR image coding system - Image coding specification," ISO/IEC International Standard 29199-2 and ITU-T Recommendation T.832 (2012).
9. 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.* **9**(8), 1309–1324 (2000).
10. R. Starosolski, "Application of reversible denoising and lifting steps to LDgEb and RCT color space transforms for improved lossless compression," in *Proc. BDAS 2016, Springer CCIS*, Vol. 613, pp. 623–632 (2016).
11. R. Starosolski, "Application of reversible denoising and lifting steps to DWT in lossless JPEG 2000 for improved bitrates," *Signal Process. Image Commun.* **39**, 249–263 (2015).
12. H. S. Malvar and G. J. Sullivan, "YCoCg-R: a color space with RGB reversibility and low dynamic range," ISO/IEC JTC1/SC29/WG11 and ITU-T SG16 Q.6 Document JVT-I014r3 (2003).
13. H. Murakami, S. Matsumoto, Y. Hatori, and H. Yamamoto, "15/30 Mbit/s universal digital TV codec using a median adaptive predictive coding method," *IEEE Trans. Commun.* **35**(6), 637–645 (1987).
14. S. A. Martucci, "Reversible compression of HDTV images using median adaptive prediction and arithmetic coding," in *Proc. IEEE Int. Symp. Circuits Systems*, Vol. 2, pp. 1310–1313 (1990).
15. T. Strutz, "Adaptive selection of colour transformations for reversible image compression," in *Proc. 20th European Signal Processing Conf.*, pp. 1204–1208 (2012).
16. "Set (Colour Set) of color images from the University of Waterloo, Fractal Coding and Analysis Group," http://links.uwaterloo.ca/Repository.html (21 February 2016).
17. "Set of images from the Kodak corporation," http://www.cipr.rpi.edu/resource/stills/kodak.html (21 February 2016)
18. F. De Simone et al., "Subjective evaluation of JPEG XR image compression," *Proc. SPIE* **7443**, 74430L (2009).
19. "Set of images from École \'Ecole polytechnique f\'ed\'erale de Lausanne," http://documents.epfl.ch/groups/g/gr/gr-eb-unit/www/IQA/Original.zip (21 February 2016)
20. "Sets A1, A2, and A3 of color images in optical resolutions of acquisition devices," http://sun.aei.polsl.pl/~rstaros/optres/ (21 February 2016)
21. R. Starosolski, "Compressing high bit depth images of sparse histograms," in *Int. Electronic Conf. on Computer Science, AIP Conf. Proc.* Vol. 1060, pp. 269–272, American Institute of Physics (2008).
22. A. J. Pinho, "Preprocessing techniques for improving the lossless compression of images with quasi-sparse and locally sparse histograms," in *Proc. ICME*, Vol. 1, pp. 633–636 (2002).
23. D. S. Taubman and M. W. Marcellin, *JPEG2000 Image Compression Fundamentals, Standards and Practice*, Springer Science + Business Media, New York (2002).
24. S. Srinivasan et al., "HD Photo: a new image coding technology for digital photography," *Proc. SPIE* **6696**, 66960A (2007).
25. F. Dufaux, G. J. Sullivan, and T. Ebrahimi, "The JPEG XR image coding standard," *IEEE Signal Proc. Mag.* **26**(6), 195–199 (2009).
26. SPMG/UBC, "Signal Processing and Multimedia Group, Univ. of British Columbia JPEG-LS implementation, version 2.2," http://www.stat.columbia.edu/~jakulin/jpeg-ls/mirror.htm (21 February 2016).
27. M. Adams, "JasPer implementation of JPEG 2000, version 1.900," http://www.ece.uvic.ca/~mdadams/jasper/ (21 February 2016).
28. M. D. Adams and R. K. Ward, "JasPer: a portable flexible open-source software tool kit for image coding/processing," in *Proc. IEEE Int. Conf. on Acoustics, Speech, and Signal Processing*, Vol. 5, pp. 241–244 (2004).
29. ISO/IEC and ITU-T, "Information Technology–JPEG XR Image Coding System - Reference Software," ISO/IEC International Standard 29199-5 and ITU-T Recommendation T.835 (2012).
30. "Implementation of RDLS-modified color space transforms (colortransf-rdls), version 0.9," http://sun.aei.polsl.pl/~rstaros/colortransf-rdls/ (21 February 2016)
31. R. C. Hardie and K. E. Barner, "Rank conditioned rank selection filters for signal restoration," *IEEE Trans. Image Process.* **3**(2), 192–206 (1994).
32. T. Bernas et al., "Application of detector precision characteristics and histogram packing for compression of biological fluorescence micrographs," *Comput. Methods Programs Biomed.* **108**(2), 511–523 (2012).
33. T. Bernas, R. Starosolski, and R. Wójcicki, "Application of detector precision characteristics for the denoising of biological micrographs in the wavelet domain," *Biomed. Signal Process.* **19**, 1–13 (2015).

Roman Starosolski is an assistant professor at the Institute of Informatics, Silesian University of Technology. He received his MSc and PhD degrees in computer science from the Silesian University of Technology in 1995 and 2002, respectively. He is the author of about 30 papers and has written five book chapters. His current research interests include image processing, compression of image, video and volumetric data, color space transforms, medical imaging, and image compression standards.