Lifting-based reversible color transformations for image compression

Henrique S. Malvar, Gary J. Sullivan, and Sridhar Srinivasan
Microsoft Corporation, One Microsoft Way, Redmond, WA 98052, USA

ABSTRACT

This paper reviews a set of color spaces that allow reversible mapping between red-green-blue and luma-chroma representations in integer arithmetic. The YCoCg transform and its reversible form YCoCg-R can improve coding gain by over 0.5 dB with respect to the popular YCrCb transform, while achieving much lower computational complexity. We also present extensions of the YCoCg transform for four-channel CMYK pixel data. Thanks to their reversibility under integer arithmetic, these transforms are useful for both lossy and lossless compression. Versions of these transforms are used in the HD Photo image coding technology (which is the basis for the upcoming JPEG XR standard) and in recent editions of the H.264/MPEG-4 AVC video coding standard.

Keywords: Image coding, color transforms, lossless coding, YCoCg, JPEG, JPEG XR, HD Photo.

1. INTRODUCTION

In color image compression, usually the input image has three color values per pixel: red, green, and blue (RGB). Independent compression of each of the R, G, and B color planes is possible (and explicitly allowed in standards such as JPEG 2000[1,2] and JPEG XR[3,4]), but it is not efficient for natural images, such as pictures from digital cameras. For such images the color planes have significant correlation, and thus a linear color transform should be applied to map the RGB color space into a space that can lead to better compression. For lossless compression, the color transformation must be itself lossless, i.e., the conversion from each integer-valued RGB pixel triplet into a new integer-valued triplet in different color space must be reversible in integer arithmetic.

The most commonly used color transform is the one that maps the RGB space into the YCrCb space[1,5]. There are several definitions of the space, with different scaling factors applied to the chroma channels. One such definition is:

$$\begin{bmatrix} Y \\ Cr \\ Cb \end{bmatrix} = \begin{bmatrix} 0.299 & 0.587 & 0.114 \\ 0.5 & -0.4187 & -0.0813 \\ -0.1687 & -0.3313 & 0.5 \end{bmatrix} \begin{bmatrix} R \\ G \\ B \end{bmatrix} \tag{1}$$

which has an inverse given by

$$\begin{bmatrix} R \\ G \\ B \end{bmatrix} = \begin{bmatrix} 1 & 1.402 & 0 \\ 1 & -0.714 & -0.344 \\ 1 & 0 & 1.772 \end{bmatrix} \begin{bmatrix} Y \\ Cr \\ Cb \end{bmatrix} \tag{2}$$

The transform above maps the RGB space into a luma component Y and two chroma components, Cr and Cb. The main reason for such a mapping is that the human visual system is much less sensitive to high-frequency components in chroma. Thus, compression systems such as JPEG[6] can downsample the chroma components (usually by 2:1 in each of the horizontal and vertical directions), as well as increase their quantization step sizes with respect to luma, to achieve further compression. Note that Cr and Cb are scaled versions of $(R - Y)$ and $(B - Y)$, so Cr and Cb can be interpreted as measures of how much red and blue content in a pixel differs from luma, respectively. For a gray pixel, $R = G = B = Y$, and so $Cr = Cb = 0$.

Note that in this paper, we use the terms *luma* and *chroma* rather than *luminance* and *chrominance*. This is to avoid implying that these signals have a linear relationship with true luminance and chrominance. In fact, it is more common

Applications of Digital Image Processing XXXI, edited by Andrew G. Tescher,
Proc. of SPIE Vol. 7073, 707307, (2008) · 0277-786X/08/$18 · doi: 10.1117/12.797091

for both the RGB and luma-chroma signals discussed in this paper to involve nonlinear (gamma compensated) relationships with light intensities. Although some authors use prime symbols to indicate the presence of such nonlinear relationships, these are omitted here for notational simplicity.

Historically, the YCrCb space is closely related to the YIQ space of the NTSC television standard[7]. The matrices in (1) and (2) were derived by principal component analyses (PCA) on video data, performed decades ago (with the rotation matrix from CrCb to IQ determined to approximate the PCA results and thus minimize the required bandwidths to transmit the *I* and *Q* components), combined with a desire to have the luma channel approximate the intensity perception of the human visual system[5].

With modern high-resolution cameras producing sharper images, the question arises on whether the YCrCb is the best color space for compression[8]. We present in the Section 2 the new color space YCoCg, which has simple transformation matrices, allowing the mapping from YCoCg back to RGB to be computed with only four additions. In Section 3 we extend the design for lossless mappings under integer arithmetic, and in Section 4 we extend those results to four-channel image data in the CMYK format.

2. THE YCoCg COLOR SPACE

From a compression standpoint, the best color space is the one in which the components are uncorrelated. For a rich enough data set, we can compute the 3×3 inter-color correlation matrix, and from it compute the Karhunen-Loève transform[5] (KLT, which is closely related to PCA), which provides maximum decorrelation. We computed statistics from the 24 RGB images of size 768×512 pixels in the standard Kodak image set (available from the JPEG site), and obtained the KLT, which can be closely approximated by rational numbers*, in the form

$$\begin{bmatrix} Y \\ C_1 \\ C_2 \end{bmatrix} = \begin{bmatrix} 1/3 & 1/3 & 1/3 \\ 1/2 & 0 & -1/2 \\ -1/4 & 1/2 & -1/4 \end{bmatrix} \begin{bmatrix} R \\ G \\ B \end{bmatrix} \Leftrightarrow \begin{bmatrix} R \\ G \\ B \end{bmatrix} = \begin{bmatrix} 1 & 1 & -2/3 \\ 1 & 0 & 4/3 \\ 1 & -1 & -2/3 \end{bmatrix} \begin{bmatrix} Y \\ C_1 \\ C_2 \end{bmatrix} \tag{3}$$

The transform above was also verified experimentally on different data sets by other authors[9,10]. From it we can make a couple of observations: first, the KLT preserves the luma-chroma separation, but the luma Y is computed as simply the average of the R, G, and B values; second, one of the color channels is proportional to $R - B$ (which is proportional to $Cr - Cb$), and the other proportional to $G - (R + B) / 2$, that is, the difference between G and the average of R and B.

The form of the inverse color mapping in (3) inspires the question: can the inverse transform be made computationally even simpler? One easy positive answer to that can be obtained by simply rounding all entries of the original inverse KLT transform to the nearest integer. That leads to the YCoCg color space definition[11]

$$\begin{bmatrix} Y \\ Co \\ Cg \end{bmatrix} = \begin{bmatrix} 1/4 & 1/2 & 1/4 \\ 1/2 & 0 & -1/2 \\ -1/4 & 1/2 & -1/4 \end{bmatrix} \begin{bmatrix} R \\ G \\ B \end{bmatrix} \Leftrightarrow \begin{bmatrix} R \\ G \\ B \end{bmatrix} = \begin{bmatrix} 1 & 1 & -1 \\ 1 & 0 & 1 \\ 1 & -1 & -1 \end{bmatrix} \begin{bmatrix} Y \\ Co \\ Cg \end{bmatrix} \tag{4}$$

This new transform has two advantages over the approximate KLT:

(i) the influence of the green channel on the luma channel is stronger, which is desirable from a visual perception consideration;

(ii) the inverse mapping from YCoCg to RGB has just additions (or subtractions). In fact, it can be computed with only four additions, in the form: $G = (Y + Cg)$; $t = (Y - Cg)$; $R = t + Co$; $B = t - Co$.

* Note that the rational approximation in (3) keeps the synthesis basis functions (the columns of the inverse transform matrix) orthogonal, but their squared norms are now different (3, 2, and 8/3, respectively), so that in a compression application the step sizes for each of the channels would have to be adjusted accordingly.

The chroma channels in (4) satisfy $Cg = G - Y$ and $Co = (R - Y) + (G - Y)$. So, Cg is large when G is large relative to Y, i.e. for green pixels, and Co is large when both R and G are large relative to Y, i.e. for orange pixels. Thus, chroma channels *Co* and *Cg* in (4) can be viewed as *excess orange* and *excess green*, respectively, and hence the name YCoCg.

Another relatively simple color space transformation is the reversible color transform[12] (RCT) of JPEG 2000, whose linear matrix equivalent is given by

$$\begin{bmatrix} Y \\ Cu \\ Cv \end{bmatrix} = \begin{bmatrix} 1/4 & 1/2 & 1/4 \\ 0 & -1 & 1 \\ 1 & -1 & 0 \end{bmatrix} \begin{bmatrix} R \\ G \\ B \end{bmatrix} \Leftrightarrow \begin{bmatrix} R \\ G \\ B \end{bmatrix} = \begin{bmatrix} 1 & -1/4 & 3/4 \\ 1 & -1/4 & -1/4 \\ 1 & 3/4 & -1/4 \end{bmatrix} \begin{bmatrix} Y \\ Cu \\ Cv \end{bmatrix} \tag{5}$$

A metric of compression performance is the transform coding gain, defined by the ratio of the arithmetic mean to the geometric mean of the variances of the variables in the new coordinates, properly scaled by the norms of the synthesis basis functions for nonunitary transforms[13]. The coding gain is usually measured in dB, representing the reduction in quantization noise by quantizing in the transformed domain instead of the original domain[13,14]. In Table 1 we show the coding gains for the color spaces YCrCb and YCoCg, as well as for the RCT transform, for the statistics computed from the Kodak image set. We see that the RCT transform has a gain of 0.44 dB over the usual YCrCb space, and YCoCg has a gain of 0.67 dB over YCrCb, that is, an additional 0.23 dB over the RCT. That is a modest gain, but it is in fact a somewhat surprising result. By making the inverse transform as simple as possible, to minimize computational complexity, we would expect a penalty in coding gain, but in fact the YCoCg has a better coding gain than both the YCrCb and the RCT.

Color Transform	Coding gain, dB
Optimal (KLT)	4.54
KLT approximation (eqn. 3)	4.42
ITU-R BT.470	3.54
JPEG 2000 RCT	3.98
YCoCg	4.21

Table 1 – Coding gain for various 3-channel color-space transforms, for images in the Kodak test set.

In practice, the coding gain improvement may be lower than the values on Table 1. First because those are asymptotic numbers for high bit rates[13,14], second because image codecs usually downsample the color planes, reducing the coding gain from the color transform, and third because the particular image being encoded may have statistics that are significantly different from the ensemble statistics of the Kodak test set. Still, Table 1 is an indication that it would be difficult to find a color space with better coding performance than YCoCg, since its coding gain is close to the optimal provided by the KLT (which is signal-dependent).

We modified a JPEG codec to use the transforms above and tested it on the Kodak set. We found that at high quality settings the RCT led to an improvement in peak signal-to-noise ratio (PSNR) over the YCrCb space of about 0.15 dB, and the YCoCg had an additional 0.10 dB improvement. The PSNRs were computed in the original RGB domain. Subjectively, pictures encoded with the different color spaces looked similar, except that YCoCg and RCT led to fewer artifacts in blue areas (e.g. sky). We believe that happens because the gain from *Cb* to *B* in the YCrCb inverse transform (2) is relatively large, so that quantization errors in blue areas are amplified.

Another performance metric is computational complexity. Both the RCT and the YCoCg transforms have the advantage over YCrCb that no multiplications are necessary for either the direct or inverse transform. That's because multiplications by factors such as ½ or ¼ can be implemented via right shifts. By rearranging the order of operations in (3)–(5) and introducing intermediate variables, we can compute the direct RCT with 4 additions and 2 shifts, and the inverse with 4 additions and 1 shift. The YCoCg transform is a bit more efficient: the direct transform takes 4 additions and 1 shift, and the inverse takes just 4 additions.

We see that the YCoCg space is an improvement over the RCT space and a significant improvement over the YCrCb space, both in terms of coding gain and complexity. So, the YCoCg space can be a good alternative for new designs.

3. LOSSLESS TRANSFORMATIONS AND YCoCg-R

An advantage of the RCT in (5) over YCrCb is that it can easily be converted to a lossless integer transform by[1,12]

$$\begin{aligned} Y &= \left\lfloor \frac{R+2G+B}{4} \right\rfloor & G &= Y - \left\lfloor \frac{Cu+Cv}{4} \right\rfloor \\ Cu &= R-G & \Leftrightarrow \quad R &= Cu+G \\ Cv &= B-G & B &= Cv+G \end{aligned} \tag{6}$$

where $\lfloor \cdot \rfloor$ denotes truncation to the nearest integer from below (the "floor" operator).

The RGB ↔ YCoCg transform can also be made lossless by using the common *lifting* technique[15] of converting a +1/–1 butterfly into a "truncated mean"/difference butterfly (also known as S-transform[16] or modified Haar transform[17]), because the latter can be exactly inverted in integer arithmetic. For example, given two integer numbers x and y, their integer difference d and truncated average m can be computed by the formulas

$$\begin{aligned} d &= x-y & & y = m - \lfloor d/2 \rfloor \\ m &= y + \lfloor d/2 \rfloor & \Leftrightarrow \quad & x = y+d \end{aligned} \tag{7}$$

which also indicate how to recover x and y from d and m. Using (7) we can convert the RGB ↔ YCoCg transform in (4) to lossless by first scaling the Co and Cg components by a factor of two, so the S-transform in (7) can be directly applied. The final result is given by the equations below, which are also depicted in Figure 1.

$$\begin{aligned} Co &= R-B & t &= Y - \lfloor Cg/2 \rfloor \\ t &= B + \lfloor Co/2 \rfloor & G &= Cg+t \\ Cg &= G-t & \Leftrightarrow \quad B &= t - \lfloor Co/2 \rfloor \\ Y &= t + \lfloor Cg/2 \rfloor & R &= B+Co \end{aligned} \tag{8}$$

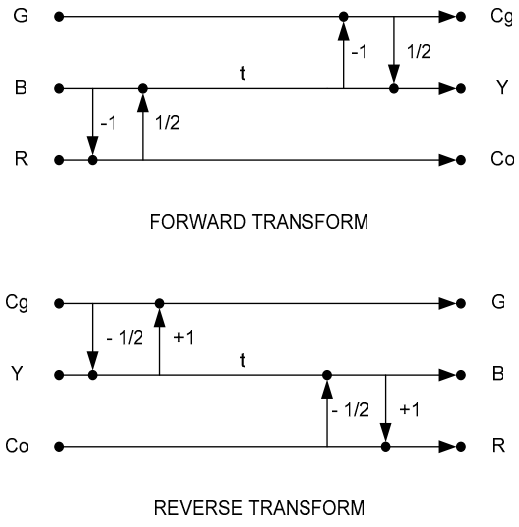

Figure 1 – Flowgraph of the reversible form of the RGB ↔ YCoCg-R color mapping. The luma output Y has the same dynamic range as the *RGB* channels, and *Co* and *Cg* require only one additional bit of dynamic range, without loss of coding gain performance when compared to the original YCoCg.

Note that the operator $\lfloor \cdot/2 \rfloor$ corresponds to a right shift. By rolling out the equations for the direct transform in (8), we see that the *Co* and *Cg* channels are scaled by a factor of two when compared to the original definition in (4), namely

$$Co = R - B, \quad Cg \cong G - \frac{R+B}{2}, \quad Y \cong \frac{G}{2} + \frac{R+B}{4} \tag{9}$$

In the form above, we see that the direct transform matrix (which maps RGB to YCoCg) has a determinant whose magnitude equals one, which is a necessary condition for optimal lossless compression performance. It is not possible to design a reversible transform matrix whose determinant has a magnitude lower than one; if it is higher than one, then dynamic range expands beyond the minimum necessary, hurting compression performance.

We see that the lossless RGB ↔ YCoCg uses 4 additions and 2 shifts to compute either the direct or inverse mappings. So, in the inverse mapping it uses one more shift than the RCT transform. That cost is likely to be insignificant for most applications, especially in view of the higher coding gain of the YCoCg transform. We also note that the lossless transform in (8) is similar to but more efficient than the one described in[18].

We refer to the reversible form of the YCoCg transform as YCoCg-R[19]. It is the color transform used for RGB data in HD Photo[4] and JPEG XR[3], and is also supported in the "fidelity range extensions" of H.264/MPEG-4 AVC.[19,20,21] HD Photo and JPEG XR implement the YCoCg transform with the slightly modified flowgraphs shown in Figure 2, which are easily seen as an equivalent realization of the flowgraph in Figure 1. The advantage of this variant is that some amount of architectural flexibility is provided by the inner lifting steps, which may be performed in either sequence. Two adders may simultaneously process the pixel in two steps.

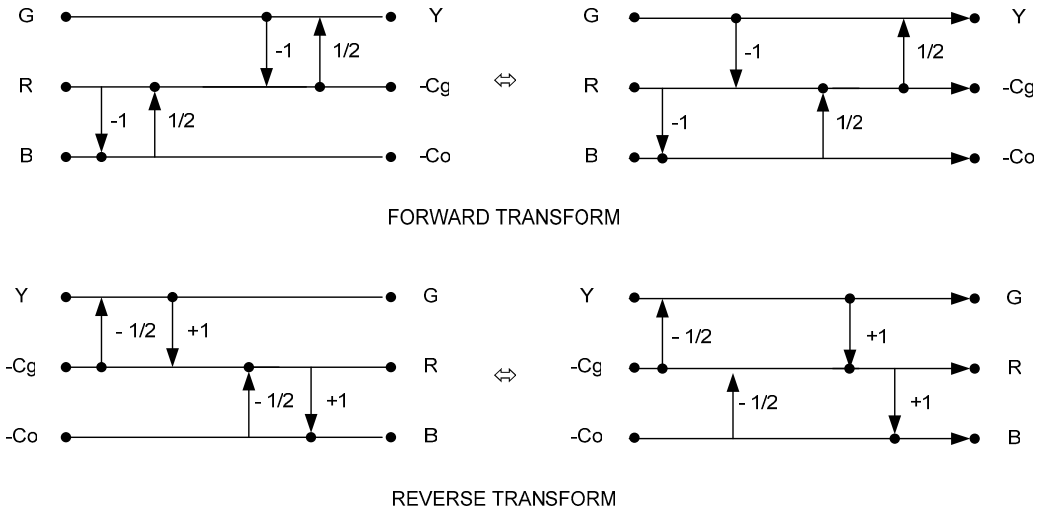

Figure 2 – Flowgraph of the reversible form of the RGB ↔ YCoCg-R color mapping used in JPEG XR. As in Figure 1, the luma output *Y* has the same dynamic range as the RGB channels, and *Co* and *Cg* require only one additional bit of dynamic range.

4. FOUR-CHANNEL REVERSIBLE COLOR TRANSFORMS

The image data to be compressed is not always in an RGB color space. Typical scenarios include the generation of four-color CMYK (cyan, magenta, yellow, and black) print image files, which can be sent to a printer or to printing services. In such cases compression of the image data is desirable, to reduce file size or network bandwidth. Thus, in this section we discuss variations of the YCoCg transform that are appropriate for CMYK data.

The CMYK format was developed for printing with four ink colors. The main advantage of specifying the black channel separately is that better rendition of black or dark tones can be directly achieved with the use of black ink, rather than

relying on a precise combination of magenta, cyan, and yellow to produce black. The CMY colors are usually the complement of RGB colors, namely[22]

$$\begin{aligned} c &= N - R \\ m &= N - G \\ y &= N - B \end{aligned} \tag{10}$$

where we assume that RGB and CMYK values are in the range [0…*N*] (for 8-bit data, $N = 255$). The *k* (black) channel is usually computed as the "minimum colorant amount", in the form[22]

$$k = f\left(\min\{c,m,y\}\right) \tag{11}$$

where the function $f(\cdot)$ depends on the printing technology and ink characteristics.

4.1 Three-Channel Transform

A first approach towards decorrelation of CMYK image data for compression is to apply a 3-channel transform to the *cmy* channels, with the aim to provide YCoCg data directly from it, in the form

$$\begin{aligned} Co &= c - y \\ t &= y + \lfloor Co/2 \rfloor \\ Cg &= t - m \\ Y' &= m + \lfloor Cg/2 \rfloor \\ Y &= N - Y' \end{aligned} \tag{12}$$

where again we have considered that the channel values are in the range [0…*N*]. We see that the *Y*′ is a 'negative luma' output, and thus *Y* is the true luma output. We see that the first four equations in (12) are quite similar to those in (8), so they can also be implemented in two lifting steps, in a structure similar to that in Figure 1.

It is easy to verify that the YCoCg channels computed by (12) are identical to those that would have been obtained if we had applied the inverse of the mappings in (10) and then applied the YCoCg transform as in (8). However, that would require inverting all three input channels, whereas with the transform in (12) we only need to invert one output channel.

For the *k* channel, we would leave it unmodified, so that the encoder would process the 4-channel representation YCoCgK, with $K = k$. A decoder residing in a printer, for example, would decode all channels, and thus recover the original *cmyk* data. Another decoder that just wants to display the image can simply disregard the *K* channel and decode RGB data from the YCoCg channels. Not using the *k* channel in the transform has the advantage that we don't need to worry about how the *k* values relate to the *cmy* values. In other words, by using the 3-color transform in (12) we have a predictable improvement in coding performance (that is, the same improvement as discussed in the previous session). However, in some applications the function $f(\cdot)$ in (11) is well known, and further improvement can be achieved by including the *k* channel in the color transform. We address that next.

4.2 Four-Channel Transforms

If the function $f(\cdot)$ in (11) has a relatively simple form, there may be significant linear statistical correlation between *k* values and *cmy* values. In that case, then a 4-color transform may lead to better compression performance. Let's assume, for example, the simplest possible form for that function, that is the identity function $f(r) \equiv r$. Then, the *k* channel is given by $k = \min\{c,m,y\}$. Although strictly speaking that is still a nonlinear relationship, we see that there will probably be significant correlation between *k* and the other channels. For example, for pixels with washed out colors, closer to gray, we have $c \cong m \cong y \cong k$. Thus, if we had a channel that represented an average of all *cmyk* channels, it would probably be advantageous from a compression perspective, since that would likely carry most of the information.

With the above in mind, we define a 4-color transform by

$$\begin{aligned} Co &= c - y \\ t &= y + \lfloor Co/2 \rfloor \\ Cg &= t - m \\ Y' &= m + \lfloor Cg/2 \rfloor \\ K &= Y' - k \\ Y &= N - \left(k + \lfloor K/2 \rfloor\right) \end{aligned} \tag{13}$$

in which the first two lifting operators produce the same *Co*, *Cg*, and *Y′* values as in (12), but a third lifting operator combines the *Y′* and *k* values, as shown in Figure 3.

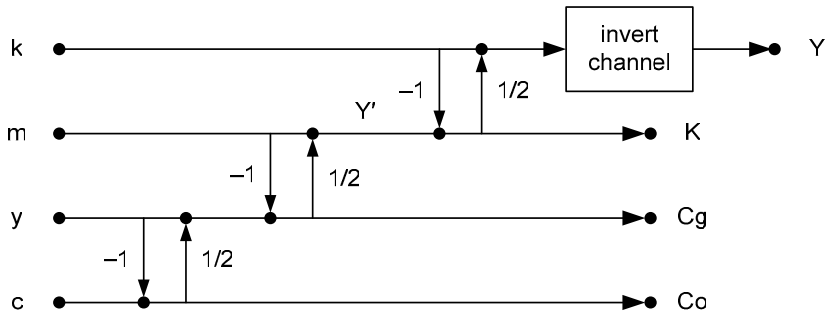

Figure 3 – Flowgraph for a reversible CMYK to YCoCgK color mapping. It has better compression performance than a 3-channel transform if channel k has been generated from a smooth function $f(\cdot)$ in (11). The inverse mapping can be easily inferred from the above, similarly to Figure 1.

Unrolling the last two equations in (13), we arrive at

$$\begin{aligned} Y &= N - \left(Y' + k\right)/2 \\ K &= Y' - k \end{aligned} \tag{14}$$

so that now *Y* carries the average of luma information from the *cmy* channels and the luma information from the *k* channel, whereas *K* carries the difference between the two.

If the function $f(\cdot)$ is very smooth, such as in the simple case $f(r) = r$, then we expect *Y′* and *k* to be reasonably close, so that *K* will have lower energy, thus further improving coding gain. Plus, *Y* will be a close approximation of the *Y* that would have been computed from (8) on the underlying RGB data, so a decoder can also generate a reasonable approximation of the RGB data from these YCoCg channel, e.g. for the purpose of image previewing. A close variant of this 4-channel transform is implemented in HD Photo and JPEG XR[4].

If approximate compatibility with YCoCg is not desired, there is a 3-stage lifting structure that can improve compression performance further. It is obtained by approximating the KLT of the 4-channel *cmyk* data[†] by simple integer-reversible operators. It has the form

$$\begin{aligned} Cx &= m - y \\ t &= y + \lfloor Cx/2 \rfloor \\ Cr &= k - c \\ s &= c + \lfloor Cr/2 \rfloor \\ Dc &= s - t \\ Y &= N - \left(t + \lfloor Dc/2 \rfloor\right) \end{aligned} \quad \Leftrightarrow \quad \begin{aligned} t &= N - \left(Y + \lfloor Dc/2 \rfloor\right) \\ s &= Dc + t \\ c &= s - \lfloor Cr/2 \rfloor \\ k &= c + Cr \\ y &= t - \lfloor Cx/2 \rfloor \\ m &= y + Cx \end{aligned} \tag{15}$$

The new chroma channels are now *Cx* and *Cr*. Note that this new *Cr* resembles the traditional *Cr* in the YCrCb, but is not the same. The traditional *Cr* is given by $Cr = R - Y$, with *Y* the luma channel as defined in (1), whereas *Cr* above satisfies $Cr = R - (N - k)$, where the term $(N - k)$ approximates a luma channel. The flowgraph for the direct transform is shown in Figure 4.

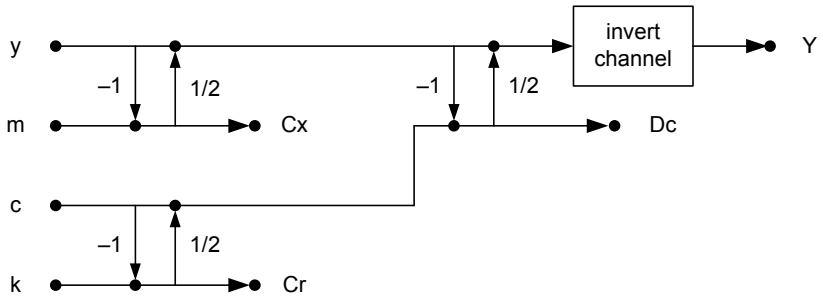

Figure 4 – Flowgraph for a reversible CMYK to YCrCxDc color mapping. It has better compression performance than the 4-channel transform in Figure 3 if channel *k* has been generated from a smooth function $f(\cdot)$ in (11). However, the output channels are not compatible with YCoCg. The inverse mapping can be easily inferred from the above, similarly to Figure 1.

In Table 2 we show the coding gain in dB of the transforms we discussed in this section for CMYK data. We used not only the 24 images of the Kodak set, as for the data in Table 1, but also the large color images from the JPEG 2000 test set (roughly 9.5 megapixels in each set). The *cmyk* channels were generated as in (10) and (11), with $f(r) \equiv r$. We see that the coding gain of the 3-channel transform is significantly lower, because it processes the *k* channel independently. However that transform is the most robust, in the sense that the coding gain is completely independent on how the *k* channel is generated. The YCoCgK transform improves the coding gain by about 2 dB, as long as the *k* channel is generated from a funciton $f(\cdot)$ that is not too distant from an identity function. Thus, the increase in coding gain comes at the expense of trading off robustness to *k*.

[†] When we refer to the KLT of *cmyk* data, we mean the KLT computed from *cmyk* images generated from the RGB images in the Kodak test set, via the mappings in (10) and (11), with $f(r) = r$.

Color Transform	Coding gain, dB Kodak set	Coding gain, dB JPEG 2000 set
Optimal (KLT)	7.39	7.14
3-channel YCoCg + k	3.14	3.53
YCoCgK	5.02	5.60
YCrCxDc	6.93	6.55

Table 2 – Coding gain for 4-channel reversible color-space transforms, for images in two test sets.

The YCrCxDc transform provides another 1–2 dB improvement in coding gain, but besides being less robust to how the *k* channel is computed, it also produces output channels that are not directly compatible with YCoCg. We see that for 4-channel image data, the use of appropriate transforms can lead to significant gains in compression performance.

We should note that in some applications the *k* information is subtracted from the *cmy* channels, in the form

$$\begin{aligned} c' &= N - R \\ m' &= N - G \\ y' &= N - B \\ k &= f\left(\min\{c, m, y\}\right) \\ c &= c' - k \\ m &= m' - k \\ y &= y' - k \end{aligned} \tag{16}$$

In that case, all final *cmyk* channels are nonlinearly related to the RGB values. The correlation between these final *cmy* channels and the *k* channel will be significantly reduced by subtraction of black from the color components (which thus makes the *cmy* channels representative of color differences). The transforms we have so far described would not be appropriate, and thus the channels should be either processed independently, or a possibly nonlinear new transform should be developed.

CONCLUSION

We presented in more detail the main ideas behind the development of the YCoCg and YCoCg-R color space transforms for RGB image data. We showed that their coding gain is higher than that of the traditional YCrCb transform, and close to that of the optimal KLT. Thus, it is unlikely that a linear color transform with simpler computational structures and better coding gains can be derived. Versions of these transforms are used in modern standards such as the draft JPEG XR image coding standard and the H.264/MPEG-4 AVC video coding standard. We also presented two constructions for 4-channel transforms, which can provide significant coding gains (5 to 7 dB) for input data in 4-channel CMYK format. A version of the YCoCgK is also used in the draft JPEG XR standard.

REFERENCES

1. D. S. Taubman and M. W. Marcellin, JPEG2000 Image Compression Fundamentals, Standards, and Practice. Boston, MA: Kluwer, 2000.
2. ITU-T and ISO/IEC, "JPEG 2000 Image Coding System," ITU-T Rec. T.800 and ISO/IEC 15444-1, March 2000.
3. ITU-T SG 16 Q.6 and ISO/IEC JTC 1/SC 29/WG 1, "Study Text for JPEG XR FCD (ISO/IEC 29199-2)", Document N4659, June 2008.
4. S. Srinivasan, C. Tu, S. L. Regunathan, and G. J. Sullivan, "HD Photo: A new image coding technology for digital photography," *Proc. SPIE Appl. of Digital Image Processing XXX*, San Diego, CA, vol. 6696, Aug. 2007.
5. W. K. Pratt, *Digital Image Processing*. New York: Wiley, 1978.

6. W. B. Pennebaker and J. L. Mitchell, *JPEG Still Image Data Compression Standard*. New York, NY: Van Nostrand Reinhold, 1993.
7. ITU-R, "Conventional Analogue Television Systems," Rec. ITU-R BT.470, 1970 (and subsequent revisions).
8. R. L. de Queiroz, "Improved transforms for the compression of color and multispectral images," *Proc. IEEE Int. Conf. Image Processing*, Rochester, NY, pp. 381-384, Sept. 2002.
9. P. Hao and Q. Shi, "Comparative study of color transforms for image coding and derivation of integer reversible color transform," *Proc. 15th Int. Conf. Pattern Recognition*, Barcelona, Spain, vol. 3, pp. 224-227, Sept. 2000.
10. S. Van Assche, W. Philips, and I. Lemahieu, "Lossless compression of pre-press images using a novel color decorrelation technique," *Pattern Recognition*, vol. 32, pp. 435-441, 1999.
11. H. S. Malvar and G. J. Sullivan, "Transform, Scaling & Color Space Impact of Professional Extensions," ITU-T/ISO/IEC JVT Document JVT-H031, available at http://ftp3.itu.ch/av-arch/jvt-site/2003_05_Geneva/JVT-H031r2.doc, May 2003.
12. M. J. Gormish, E. L. Schwartz, A. Keith, M. Boliek, and A. Zandi, "Lossless and nearly lossless compression for high quality images," *Proc. SPIE Very High Resolution and Quality Imaging II*, San Jose, CA, vol. 3025, pp. 62-70, Feb. 1997.
13. H. S. Malvar, "Biorthogonal and nonuniform lapped transforms for transform coding with reduced blocking and ringing artifacts," *IEEE Trans. Signal Processing*, vol. 46, pp. 1043-1053, Apr. 1998.
14. H. S. Malvar, *Signal Processing with Lapped Transforms*. Boston, MA: Artech House, 1992.
15. W. Sweldens, "The lifting scheme: a custom-design construction of biorthogonal wavelets," Applied and Computational Harmonic Analysis, vol. 3, pp. 186-200, 1996.
16. P. Lux, "A novel set of closed orthogonal functions for picture coding," *Arch. Elek. Ubertragung*, vol. 31, pp. 267-274, 1977.
17. P. Piscaglia and B. Macq, "Multiresolution lossless compression scheme," *Proc. IEEE Int. Conf. Image Processing*, Lausanne, Switzerland, vol. 1, pp. 69-72, Sept. 1996.
18. I. Singh, P. Agathoklis, and A. Antoniou, "Lossless compression of color images using an improved integer-based nonlinear wavelet transform," *Proc. IEEE Int. Symp. Circuits and Systems*, Hong Kong, pp. 2609-2612, June 1997.
19. H. S. Malvar and G. J. Sullivan, "YCoCg-R: A Color Space with RGB Reversibility and Low Dynamic Range," ITU-T/ISO/IEC JVT Document JVT-I014, available at http://ftp3.itu.ch/av-arch/jvt-site/2003_09_SanDiego/JVT-I014r3.doc, Sept. 2003.
20. ITU-T and ISO/IEC, "Advanced Video Coding for Generic Audiovisual Services," ITU-T Rec. H.264 and ISO/IEC 14496-10, July 2004 (and subsequent revisions).
21. G. J. Sullivan, P. Topiwala, and A. Luthra, "The H.264/AVC Advanced Video Coding Standard: Overview and Introduction to the Fidelity Range Extensions," *SPIE Conf. on Applications of Digital Image Processing XXVII (Special Session on Advances in the Emerging H.264/AVC Video Coding Standard)*, vol. 5558, part 1, pp. 454-474, Aug. 2004.
22. P. Green and L. MacDonald, *Colour Engineering*. Chichester, England: Wiley, 2002.