Color Space Considerations for Linear Image Filtering

Wilhelm Burger
Upper Austria University of Applied Sciences, Hagenberg, Austria – wilbur@ieee.org

Abstract
Filters designed for monochromatic, scalar-valued images are frequently applied to color imagery without considering the vector-valued nature of the data and the effects of nonlinear component values. This paper analyzes the implications of component-wise linear filtering upon the underlying color space and the related chromatic errors that may be unacceptable for high-quality color imaging. It suggests that perceptual uniformity and a careful choice of the working color space are important even for common smoothing filters and similar operations.

1. Introduction
Most images today are color images and filtering them is an everyday routine. Typically, linear and non-linear filters designed for grayscale images are used for color images by simply applying them to the individual color channels, i.e., by executing a scalar technique on vector-valued data. Although the resulting color artifacts appear to be acceptable for the majority of viewers, the errors introduced by these methods may be substantial and even unacceptable for high-quality color imaging. In this context, the photometric properties and, in particular, the potential non-linearities of the underlying color space are often ignored. For example, most color images produced today by digital cameras and scanners are in sRGB (“Standard RGB”) or a similar format, characterized by gamma-mapped (i.e., strongly non-linear) color components for the purpose of high coding efficiency and simplicity of use [9].

The results of linear filtering depend on the color space being used, but most image editing tools (including professional software) nevertheless apply filters directly to non-linear (sRGB) color components, without previous linearization or color space conversion. Not only does a linear operation in a non-linearly distorted space seem incorrect by intuition but it actually introduces substantial errors and visually disturbing results, as described below. Which is the correct color space to use? Should the operation be performed in linear RGB, CIEXYZ, or in a perceptually uniform color space such as CIELAB? Although the issues related to color mixtures and interpolation have been investigated for a long time (see, e.g., [4, 14]), their relevance in the context of image filtering has received only limited attention in the literature, with notable exceptions [1, 5, 10, 11].

In the following, we investigate the nature of scalar, linear filters for color images and derive requirements for the results, based on colorimetric and perceptual arguments. Four popular color spaces are evaluated: non-linear sRGB [9], linear RGB, CIEXYZ, and CIELAB [8]. We do not consider cylindrical color spaces (such as HSV or HLS) here because they lack a precise photometric specification. We first describe the implications of applying scalar linear filters to vector-based data and the implicit assumptions about the underlying color space. Next, we demonstrate how these assumptions are violated by processing images represented in non-linear and “light-linear” RGB coordinates. We subsequently illustrate that “correct” (visually acceptable) results are not obtained by relying on the physical but the perceptual properties of the working color space. Examples are shown to demonstrate these effects.
2. Scalar linear filters applied to vector-valued images

Given a discrete scalar (grayscale) image \( I(u,v) \in \mathbb{R} \), the application of a linear filter can be expressed as a linear 2D convolution

\[
\hat{I}(u,v) = [I \ast H](u,v) = \sum_{(i,j) \in R_H} I(u-i,v-j) \cdot H(i,j),
\]

where \( H \) denotes a discrete filter kernel defined over the (usually rectangular) region \( R_H \), with \( H(i,j) \in \mathbb{R} \). For a vector-valued image \( \mathbf{I} = (I_1, I_2, \ldots, I_n) \) or \( \mathbf{I}(u,v) \in \mathbb{R}^n \), with \( n \) (color) components \( I_k \), the above linear filter can be written as \( \hat{\mathbf{I}}(u,v) = [\mathbf{I} \ast H](u,v) \), with the filter kernel \( H \) still being scalar-valued. The \( k^{th} \) element of the result vector, \( \hat{I}_k(u,v) = [I_k \ast H](u,v) \), is simply the ordinary scalar convolution \( (1) \) applied to the corresponding component plane \( I_k \).

In case of a RGB color image (with \( n = 3 \) components), the filter kernel \( H \) is applied separately to the scalar-valued \( I_R, I_G, I_B \) planes, i.e.,

\[
\hat{\mathbf{I}}(u,v) = \begin{bmatrix}
\hat{I}_R(u,v) \\
\hat{I}_G(u,v) \\
\hat{I}_B(u,v)
\end{bmatrix} = \begin{bmatrix}
[I_R \ast H](u,v) \\
[I_G \ast H](u,v) \\
[I_B \ast H](u,v)
\end{bmatrix},
\]

which is how linear filters for color images are typically implemented in practice.

**Linear smoothing filters**

Let \( \mathcal{R}_{u,v} = \{c_1, \ldots, c_K\} \) denote the set of color vectors contained in the support region of the kernel \( H \) at a given position \( (u,v) \) in the original image \( \mathbf{I} \), where \( K \) is the size of \( H \). With arbitrary kernel coefficients \( H(i,j) \in \mathbb{R} \), each resulting color vector \( \hat{\mathbf{I}}(u,v) = c \) is a linear combination of original color vectors \( \mathcal{R}_{u,v} \). If the kernel is normalized, i.e., \( \sum_{(i,j)} H(i,j) = 1 \), the result is an affine combination of the original colors. In case of a typical smoothing filter, with all coefficients \( H(i,j) \) being positive and normalized, each resulting color vector \( \hat{c} \) is a convex combination of the original color vectors \( c_1, \ldots, c_K \).

Figure 1 shows the result of filtering a synthetic test image with a normalized Gaussian kernel with radius \( \sigma = 3 \). Included are the grayscale images obtained from the corresponding luminance \( (Y) \) values (see Sec. 3.). The colors inside the horizontal bar at the center of the test image are 50% desaturated but have the same luminance as their neighboring colors. The bar should thus not show in the filtered images, unless incorrect luminance values are produced. This happens in the sRGB results (Fig. 1(b)), which also show strong dark bands, particularly along the red-blue, magenta-blue, and magenta-green edges.

**Response to a color step edge**

Assume, as a special case, that the original RGB image \( \mathbf{I} \) contains a step edge separating two regions of constant colors \( c_1 = (R_1, G_1, B_1) \) and \( c_2 = (R_2, G_2, B_2) \), respectively, as illustrated in Fig. 2(b). If placed at a position \( (u,v) \), the normalized smoothing kernel \( H \) is fully supported by elements of constant color \( c_1 \) (with the kernel is not overlapping any color edge), the trivial response of the filter is \( \hat{\mathbf{I}}(u,v) = \sum c_1 \cdot H(i,j) = c_1 \cdot \sum H(i,j) = c_1 \cdot 1 = c_1 \), i.e., the unmodified, original color \( c_1 \). Alternatively, if the filter kernel is placed at some position \( (u',v') \) on the step edge, some of its coefficients \( (\mathcal{R}_1) \) are supported by pixels with color \( c_1 \), while the remaining coefficients \( (\mathcal{R}_2) \) overlap pixels with color \( c_2 \). Since \( \mathcal{R}_1 \cup \mathcal{R}_2 = \mathcal{R} \) and the kernel is normalized, the resulting color at position \( (u',v') \) is

\[
c'_k = s_1 \cdot c_1 + s_2 \cdot c_2 = s_1 \cdot c_1 + (1-s_1) \cdot c_2 = c_1 + \lambda \cdot (c_2 - c_1),
\]

where

\[
s_1 = \frac{\int_{\mathcal{R}_1} H(i,j) \, di \, dj}{\int_{\mathcal{R}} H(i,j) \, di \, dj},
\]

\[
s_2 = 1 - s_1.
\]
Figure 1. Gaussian smoothing performed in different color spaces. Synthetic test image (a); Gaussian smoothing filter applied to sRGB (nonlinear) color components (b), RGB (linear) components (c), CIELUV (d) and CIELAB components (e). The bottom row shows the corresponding luminance (Y) images. Images are available at electronically at http://staff.fh-hagenberg.at/burger/. Note that the final appearance depends strongly on display or printer characteristics and some of the effects may not show as described.

Figure 2. Discrete 2D Gaussian filter kernel with radius 3 and support region \( R \) (a). Filter kernel positioned over a region of constant color \( c_1 \) and over a color step edge \( c_1 \rightarrow c_2 \), respectively (b). Different results are obtained by linear interpolation in a “distorted” color space (c).

for some \( \lambda \in [0, 1] \), which is a point on the straight line segment between the color coordinates \( c_1, c_2 \). Thus, at a step edge between two colors, the transient colors produced by a (normalized) smoothing filter at different positions always correspond to points on the straight line between the two original color vectors. As an example, Fig. 3(a) shows the results of linear interpolation between pure red and green, calculated in four different color spaces. Note that, although the individual RGB components run over the same range of values, the actual colors produced differ strongly (also see Fig. 4(d)).

This means that, in general, the intermediate colors produced by a linear smoothing filter are contained within the convex hull of the set of contributing colors in the working \( n \)-dimensional color space, regardless which color space is used. The relationship between linear, scalar convolution and convex linear combinations thus implies that the underlying color space is metric and the triangle inequality \( \|c_1 + c_2\| \leq \|c_1\| + \|c_2\| \) holds under the Euclidean norm. The question, if the intermediate chromaticities produced by component-wise filtering are “correct”, cannot be answered from a mathematical but only from a perceptual point of view. The CIELUV and CIELAB color models were explicitly designed to provide perceptual uniformity over a large color range and facilitate the use of the Euclidean norm as a valid distance measure between
Since \( \beta \) image with the same kernel should be identical to the result of filtering only the (scalar) brightness values of the original \( R, G, B \) \( Y \) in CIEXYZ space and is thus relevant in colorimetric terms. Since "luminance" and "luma", respectively.

As a result, when a smoothing filter is continuously moved across a step edge separating two \( \beta \) with weights weighted sum of the component values, it is easy to show that brightness values of the original colors in the local region the resulting brightness \( \beta \) \( \cdot \) \( \vec{w} \).

\[ \beta(\vec{I} \ast H) \equiv \beta(\vec{I}) \ast H. \]  

(4)

Since \( \beta(\vec{I}) \ast H \) is again a convex linear mixture of the original (scalar) brightness values \( \beta(\vec{I}) \), the resulting brightness \( \beta(\vec{I} \ast H) \) in the filtered color image at any point \( (u, v) \) is bound by the brightness values of the original colors in the local region \( R_{u,v} \), that is

\[ \min_{(i,j) \in R_{u,v}} \beta(\vec{I}(i,j)) \leq [\beta(\vec{I}) \ast H](u, v) \leq \max_{(i,j) \in R_{u,v}} \beta(\vec{I}(i,j)). \]  

(5)

As a result, when a smoothing filter is continuously moved across a step edge separating two colors \( \vec{c}_1, \vec{c}_2 \), the resulting transient brightness values must increase (or decrease) monotonically between \( \beta(\vec{c}_1) \) and \( \beta(\vec{c}_2) \).

Whether the condition in (4) is satisfied depends on the definition of the color brightness function \( \beta() \). It holds in the (common) case of brightness being defined as a linear combination of the component values in the given color space. For example, brightness of linear and non-linear RGB colors \( \vec{c} = (R, G, B) \) and \( \vec{c}' = (R', G', B') \), respectively, is usually calculated as a weighted sum of the component values,

\[ Y(\vec{c}) = \vec{w} \cdot \vec{c} = w_R R + w_G G + w_B B \quad \text{or} \quad Y'(\vec{c}') = \vec{w}' \cdot \vec{c}' = w'_R R' + w'_G G' + w'_B B', \]  

(6)

with weights \( w_k, w'_k > 0 \) and \( \sum \vec{w} = \sum \vec{w}' = 1 \). The quantities \( Y \) and \( Y' \) in (6) are referred to as "luminance" and "luma", respectively.\(^2\) The luminance \( Y \) is also the corresponding coordinate in CIEXYZ space and is thus relevant in colorimetric terms. Since \( Y(\vec{c}) \) and \( Y'(\vec{c}') \) are both linear, convex combinations of the underlying component values, it is easy to show that

\[ Y(\vec{I} \ast H) \equiv Y(\vec{I}) \ast H \quad \text{and} \quad Y'(\vec{I} \ast H) \equiv Y'(\vec{I}) \ast H \]  

(7)

\(^2\)These terms are often confused or used incorrectly [7]. Surprisingly, ITU Rec. 709 [2] specifies the same weights \( \vec{w} = \vec{w}' = (0.2126, 0.7152, 0.0722) \) to be applied for both luminance and luma calculations.
hold, as required in (4). In other words, luminance is preserved by filtering a linear RGB image $I$, while luma is preserved when filtering a non-linear (sRGB) image $I'$. However, luminance is not preserved when applying a linear smoothing filter to non-linear sRGB image $I'$ and vice versa, i.e.,

$$Y(I' * H) \neq Y(I') * H \quad \text{and} \quad Y'(I * H) \neq Y'(I) * H.$$  

(8)

In the colorimetric color models CIELUV and CIELAB, brightness of a color tuple is directly expressed by the value of its $L^*$ (lightness) component, roughly defined as $L^* \approx Y^{1/2}$, where $Y$ is the corresponding luminance and CIEXYZ-coordinate, as given in (6). Since $L^*$ is a separate (scalar) vector component, linear filtering an image $I_{CIE} = (L^*, \cdot, \cdot)$ in either CIELUV or CIELAB encoding naturally preserves this quantity, i.e.,

$$L(I_{CIE} * H) \equiv L(I_{CIE}) * H.$$  

(9)

Again, lightness is not preserved if the filter operation is performed in linear RGB or sRGB space. Filtering in any of the investigated color spaces thus preserves a particular brightness quantity: luma ($Y'$) for sRGB, luminance ($Y$) for linear RGB, and lightness ($L$) for CIELUV and CIELAB.

The example in Fig. 1(b–e) shows the luminance images obtained after applying a Gaussian filter in each of the four color spaces. Most notable are the dark bands in the sRGB result at the boundaries between highly saturated color regions (Fig. 1(b)). The horizontal bar of constant-luminance colors clearly stands out. Also note that, for sRGB, the brightness function inside these dark bands can even become non-monotonic, as illustrated by the solid $Y$-curve in Fig. 3(a) for the transition from red to green. If we take lightness ($L^*$) as the quantity that best matches human brightness perception, the question is if filtering in linear RGB or sRGB can be used as a substitute. As shown in Fig. 3(a), the luminance obtained from interpolation between red and green in linear RGB is close to the corresponding CIELUV/CIELAB curve (also seen in the resulting color shades in Fig. 4(a–c)). However, the interpolation between black and white in Fig. 3(b) shows that the luminance from linear RGB deviates strongly from the CIE models. In this case, sRGB would actually be the better choice—which is no surprise, since sRGB gamma mapping is similar to the one used for the CIE $L^*$ component. Note that black-to-white interpolation in any color space results in the same range of gray values, but these are placed at different spatial positions in the filtered results. This explains, for example, why the result from linear RGB filtering is biased toward brighter shades around gray (hueless) regions than any of the other methods (see the square markers in Fig. 1(c)). Thus neither sRGB nor linear RGB are suitable models for linear filtering from a brightness-preservation point of view.

4. Transient colors and gamut limitations

In addition to preserving the overall brightness of an image, we are interested in the transient chromaticities that are introduced by linear filtering. As illustrated in Figs. 3 and 4, interpolation in different color spaces may result in massive variations of the individual RGB components and the resulting composite colors (Fig. 4(d)). Again it seems reasonable to use the perceptually uniform CIE models as a reference. Both the CIELUV and CIELAB color spaces are specified by nonlinear transformations to and from the CIEXYZ reference space to obtain

\footnote{See [8] for precise definitions.}
Figure 4. Transient chromaticities obtained by uniform linear interpolation in different color spaces. Interpolation between high-chroma colors red/green (a), red/magenta (b), and yellow/blue (c) and the corresponding luminance ($Y$) values (e–g). Note the non-monotonic luminance from sRGB in (e, f). (d) shows the color trajectories in linear RGB space for the yellow/blue transition in (c). Results of black/white interpolation (h).

Figure 5. Uniformly tesselated sRGB cube mapped to CIEXYZ (a), CIELUV (b) and CIELAB (c) color spaces. Note that out-of-gamut colors may occur in any of the perceptually uniform CIE spaces due to the non-convexity of the corresponding volumes.

While the lightness component ($L^*$) is defined identically in CIELUV and CIELAB, their chroma components are defined differently. In CIELAB, the $a^*$, $b^*$ components are color differences that are also gamma-mapped, i.e., non-linearly related to differences in XYZ space. This is different in CIELUV, where the $u^*$, $v^*$ components are obtained by a central projection from $X$- and $Z$-coordinates, respectively. Although this is also a non-linear transformation, straight lines in CIELUV space map to straight lines in the CIE chromaticity diagram such that, at least for a constant luminance $Y$, linear interpolation in CIELUV will not create new colors outside the gamut volume.

4CIELAB was primarily designed for applications dealing with reflective and transmissive products, such as the graphic arts industry, while CIELUV aims at emissive color applications, including video, television broadcasting, and computer graphics.
the convex hull of the interpolated colors. While this may be considered a minor advantage for CIELUV over CIELAB, both color spaces were designed for short-range uniformity but not for long-distance color interpolation over remotely placed coordinates in the color space. Thus, even for CIELUV and CIELAB we cannot say that the perceptually “best” intermediate colors are found on a straight line between the original color points, as implied by the linear filter. Another concern is if the interpolated colors are actually contained in the gamut volume of the original color space. Clearly, no out-of-gamut colors can result from a convex combination of color vectors in sRGB or linear RGB, because the hull of both color spaces is convex (Fig. 5(a)) by definition. Any convex combination of colors results in a valid vector inside the working color space. However, transformed to CIELUV or CIELAB, the set of possible sRGB or linear RGB colors forms a non-convex shape, as illustrated in Fig. 5(b, c). A linear interpolation in any of these spaces may thus result in a color that is not contained inside the original RGB gamut. Particularly critical in both CIELUV and CIELAB are the bright red-to-white, red-to-yellow and red-to-magenta regions, as well as high-chroma yellow-to-green in CIELAB, where the resulting distances from the hull can be substantial (Fig. 6). Also note that in Fig. 3(a) the blue component for CIELAB interpolation is slightly negative and thus outside the RGB gamut.

Correcting out-of-gamut errors is a non-trivial problem, usually faced in the context of color management and gamut mapping. Unfortunately, the classic algorithms for gamut mapping [6] are probably too complex to warrant their use in linear filtering, while clipping individual out-of-gamut components to their admissible range can potentially cause color artifacts. Fortunately, out-of-gamut colors appear infrequently with practical images and only from extremely saturated colors (e.g., around specular highlights), such that the errors due to component clipping are usually tolerable. In terms of maximizing image quality this is nevertheless a suboptimal solution.

5. Summary and implications

In the above we have shown that applying a linear filter to the individual components of a color image presumes a certain “linearity” of the underlying color space and smoothing filters implicitly perform linear mixing and interpolation. Despite common practice, it thus appears particularly unreasonable to perform such an operation directly on gamma-mapped sRGB color channels and the poor results demonstrate that this is indeed the case. However, unlike one may expect, filtering in linear, light-proportional RGB instead does not yield better overall
results and is thus no viable option either. In summary, both (non-linear) sRGB and (linear) RGB are unsuitable for linear interpolation in order to produce perceptually accurate results. CIELUV as well as CIELAB are good choices for linear interpolation because of their metric properties, where CIELUV may be the slightly better option regarding large color distances. However, both CIE models may produce out-of-gamut chromaticities at strong color edges. Although the focus of this paper is on linear smoothing filters, similar considerations apply to other types of filters, such as linear interpolation filters for geometric image transformations, decimation filters for multi-resolution techniques, or gradient filters for color edge detection, and also non-linear filters, such as the vector median filter [3,12,13]. While complex color space transformations (e.g., sRGB ↔ CIELUV) are usually avoided in filtering for efficiency reasons, this should be strongly considered when high-quality results are important. The computational overhead appears to be affordable with modern hardware and the mappings can easily be tabulated for real-time calculation.

References