Prelinearization Stages on Color-Management Application-Specific Integrated Circuits (ASICs)

Marti Maria, Hewlett-Packard

Abstract

This paper presents a method to compute a linearization stage that may increase accuracy on lookup based RGB to RGB color transforms as implemented in Application Specific Integrated Circuits (ASIC). While the use of a prelinearization stages is a well know topic (1), there is few literature on how to build such curves. The method presented here is accurate, computationally efficient and cheap in terms of implementation. It is especially effective when apparent gamma of input and output RGB spaces are quite different. On the paper, the effect of this algorithm on linearization stage in RGB to RGB color transforms is evaluated theoretically and experimentally though psychophysical testing using real images. Both theory and the test results confirm a significant performance increase in terms of image quality.

Introduction

An ASIC (application-specific integrated circuit) is a microchip designed for a special application, such as a particular kind of transmission protocol or a hand-held computer. ASICs are used in a wide-range of applications, including color management on printers, environmental monitoring, and personal digital assistants (PDAs).

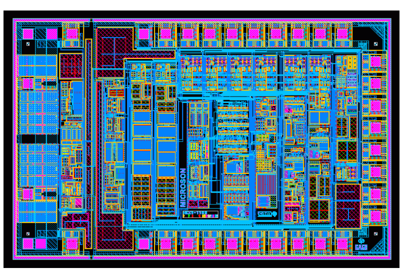


Figure 1, NEXO ASIC

An ASIC can be pre-manufactured for a special application or it can be custom manufactured (typically using components from a "building block" library of components) for a particular customer application. One of the uses of ASICs is color management. In this case the ASIC may take care of color transforms.

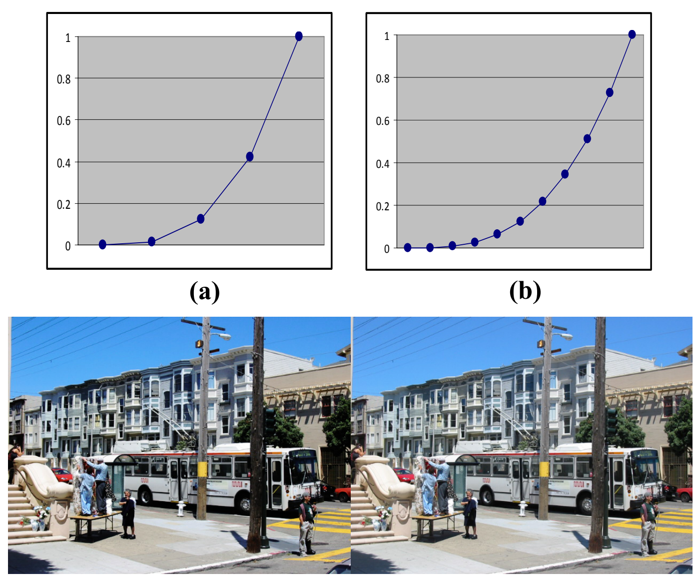
When dealing with RGB to RGB color transforms, many implementations of color-management ASIC rely on 3D look up table (LUT) with interpolation (1). Yet, although the result of 3D LUT is in general accurate, obtaining proper accuracy when source and destination spaces are very different requires a high number of grid nodes. ASIC devices usually have severe constraints on resources, so alternative solutions should be considered.

One easy method to increase apparent resolution is the use of prelinearization stages. Such constructs are already available on hardware in some of the latest generation color-management ASICs. But computing the curve to be applied by means of traditional methods is often non-optimal and very time consuming. Having an algorithm which is both, fast and accurate to compute those curves is critical in environments where the printer is to be calibrated online.

The problem

When using 3D lookup table with interpolation, nonlinear color transformations have a profound influence on the precision requirements, because they modify the relative spacing of quantization levels (2). If input RGB is quantized to a number of equally spaced perceptual levels, these will be mapped to unequally spaced perceptual levels by gamma correction; some of those levels, particularly in the highlights, may be so closely spaced that they are imperceptible.

The issue comes when apparent gamma of both color spaces is quite different and the number of available grid nodes is small. This is readily evident when transforming a continuous gray gradient across 3D LUTS of 6 and 17 grid points. Let's take for example a RGB to RGB transform going from camera RAW (apparent gamma 1.0) to sRGB (apparent gamma 2.2)



(a) (b)

(c) (d)

Figure 2, *Quantization effects generated by a 6 point 3D look up table. On increasing the number of grid points to 17 (b), the exponential curve becomes smoothed. This is very evident on images as blocking and loss of shadow detail (c and d).*

Apparent gamma

Many RGB color spaces are non-linear on respect to CIEXYZ colorimetric space. However, it is quite frequent this non-linearity has channel independence. In this way, a RGB color space may be described as a non-liner tone curve, which takes place at each R, G and B channels, and then an approximately linear part, with a 3x3 matrix multiplication to transform from one primary system to another. This is certainly the case of display monitors and synthetic workspaces like sRGB, Adobe RGB or Apple RGB. Other RGB spaces, like RAW on digital cameras and native device space for some printers have been accommodated to those constrains just because that makes such spaces easier to manipulate.

One of the first devices to use natively RGB was CRT displays. Traditionally, the CRT response curve is plotted as a power function, and the power is called gamma. One common model for CRT displays is the Gamma-gain-offset model, described by Berns (3).

$$R_{Scalar} = \left[gain_r \left(\frac{dc_r}{dc_{r,\max}} \right) + offset \right]^{\gamma} \quad (1)$$

Because this heritage, we could therefore compare any RGB space with a CRT analog, and talk about “apparent gamma”, which would be the exponent used if the RGB space would be fitted on a CRT model. Certainly most parameters would make no sense at all on many RGB spaces, because the true nature of the involved device does not correspond to a CRT display, but this fitted gamma would still give important information about the order of the space. In this way, we could assign an apparent gamma 1.0 to CIEXYZ and an apparent gamma of 2.4 to CIE L*a*b*

Modeling the color transform

Let’s assume we have function T describing the RGB → RGB’ color transform. This function is not mean to be computationally efficient and may be implemented by using device models, regression or any other methodology (1). We could therefore, evaluate any RGB value across this transform and obtain the corresponding RGB’ triplet.

$$\begin{pmatrix} R' \\ G' \\ B' \end{pmatrix} = T \cdot \begin{pmatrix} R \\ G \\ B \end{pmatrix} \quad (2)$$

According to a recent study (4), CIEXYZ values can be obtained for each RGB channel by interpolating the tristimulus values measured for the corresponding primary RGB values, plus the channel-interdependency. This assumption can be used to predict the CIE tristimulus values for any arbitrary combination of RGB values. If the channel interdependency is small enough, we can decouple the non-linearity of the color transform in a set of independent functions (*fr*, *fg* and *fb*) plus a linear transformation (M).

$$\begin{pmatrix} R' \\ G' \\ B' \end{pmatrix} = T \cdot \begin{pmatrix} R \\ G \\ B \end{pmatrix} = M \cdot \begin{pmatrix} fr(R) \\ fg(G) \\ fb(B) \end{pmatrix} \quad (3)$$

fr, *fg* and *fb* are implemented as prelinearization curves and M as the 3D LUT. Since M is not only a plain 3x3 matrix but a 3D LUT, is still able to deal with channel interdependency, as long as it is small enough. At that point is important to note than using linearization curves means a resampling on M. The proposed method takes care of that resample when populating the LUT, and the computational cost of this operation is very low.

Proposed solution

In this section we describe our proposed approach to calculate the prelinearization curves. The algorithm relies on three basic premises that are true on many common RGB spaces:

- R=G=B represents a neutral value respect to space’s white point.
- Apparent gamma of each component is approximately decoupled of other components.
- Both RGB spaces are smooth.

Despite the algorithm is not expected to work on RGB spaces out of those requirements, it has been found to perform well if characteristics are close enough. That is, an RGB space on which gamma is not completely decoupled, may also work if the interaction of channels is not strong close to neutral axis. In any case, a non-suitable conversion is detected by the algorithm and whole linearization stage is then discarded.

The proposed method includes three steps. First step builds candidate linearization curves. Second step validates those curves, and third step resamples the 3D LUT to calculate values for each lattice point.

Tone curve construction.

For obtaining the tone curve for each RGB component, a number of RGB triplets are evaluated across the color transform. In 8-bit RGB transforms, 256 values are used. In 16-bits color transforms, a maximum of 4096 values was found to be enough.

$$\begin{pmatrix} R_0' \\ G_0' \\ B_0' \end{pmatrix} = T \cdot \begin{pmatrix} \tau \\ \tau \\ \tau \end{pmatrix} \qquad \tau = R = G = B \quad (4)$$

The input values are meant to be near neutral, with (*R*=*G*=*B*). The obtained values are not necessarily neutral as *R0*, *G0* and *B0* components may differ due to channel interdependency. This generates 3 sets of values ranking from zero to white point.

Those points are then fitted and smoothed into curves by using finite difference smoother (5). The algorithm is very well suited to discrete smoothing and interpolation. A combined measure of fit and smoothness is:

$$S = \sum_{i=1}^{m} w_i (y_i - z_i)^2 + \lambda \sum_{i=n+1}^{m} (\sum_{j=0}^{n} a_j z_{i-j})^2 \qquad (5)$$

Where lambda is a parameter by which we can trade smoothness of z against fit to data y. Some of the data points y_i may be missing, and are handled across weight w_i. Normally $w_i = 1$, but when y_i is missing, the corresponding $w_i = 0$ and we can give any value we like to y. To minimize S, we set partial derivatives $\delta S/\delta z_i$ equal to zero and obtain a system of m equations. A simple adaptation of Cholesky decomposition (6) is used to solve the system. In order to assure endpoints, values for black and white are fixed in 0 and 255, so the smoother is not allowed to move those points.

Next, the lowest part of the curve is replaced by a slope-limiting linear part in order to get rid of noise and high slope situations. The cutoff is placed at 2% of the curve, which only affects digital counts 0...6 in 8 bit. This linear tram has proven to be very effective on zones where monotonicity is hard to obtain. This gives us 3 smoothed tone curves *frs*, *fgs* and *fbs* which will be used as prelinearization curves.

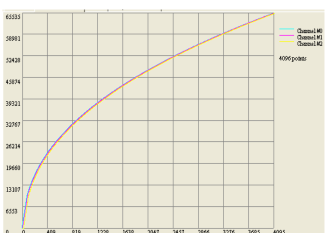


Figure 3, *obtained curves*

Tone curve validation

It is very important to have a way to automatically validate the tone curves, as wrong curves may be very destructive on terms of accuracy, and may introduce unwanted artifacts like blocking or contouring. Some checks are performed on the obtained curves, if they fail, a substitution mechanism would take place. The tests performed are:

- Check for identity curves. Each point of the curve is compared against an identity line. If most values (80%) fall within a range of 1-3 contone values of identity, then whole tone curve is discarded as it would have no effect on final transform. Slope-limiting step is not checked on this test.
- Check for non-monotonic curves. Monotonicity is required in almost any kind of color transforms. Slightly non-monotonic curves are possible due to smoothing artifacts, so it is quite important to get rid of those parts. Non-monotonic curves are not reversible and do not work with this algorithm.
- Check for Endpoints. Curve should begin at zero and end at pure white, 255 on 8-bit and 65535 on 16-bit. While failure to do so is not necessarily an error state, it may be an indication of other kind of discrepancies.

If any of those tests fails, a backup strategy is used. A statistical check is then performed on original set of points to infer whatever a simple exponential may be used. The method involves computing the average and standard deviation of ***ln(y)/ln(x)*** being y the value obtained on the tone curve and x the value used as input. Lower 7% is ignored for same reasons of slope-limiting. If standard deviation is < 0.7, then the curve is meant to be approximately an exponential, and then a pure exponential (with slope-limiting) is being used instead.

If neither method succeeds, then the RGB to RGB color transform is not suitable to be optimized by means of tone curve and is up to the user to use a different strategy, which may be to use more grid points for 3D LUT or just mark the whole transform as non optimizeable.

Populate the 3D grid

Third and last step involves the packaging part. In order to compute the values for each node, the smoothed tone curves are inverted.

$$\begin{pmatrix} frs^{-1}(R') \\ fgs^{-1}(G') \\ fbs^{-1}(B') \end{pmatrix} \qquad (6)$$

Since tone curve validation has assured monotonicity, reversing curves is an easy task. In the case of exponential fitting, the curve may be analytically inverted in both trams, slope-limiting and exponential parts. A more general case may be handled by a simple binary search, since those curves are quantized to 8 or 16 bits. Whole RGB input space is then sampled at regular intervals. Those intervals conform the indexing of RGB input space. Each one of those indexing RGB triplet is then linearly interpolated across inverse tone curves and the resulting values are evaluated across color transform. This effectively undoes any effect the curve would have. Obtained R'G'B' is used to populate the lattice point.

$$\begin{pmatrix} R' \\ G' \\ B' \end{pmatrix} = T \cdot \begin{pmatrix} fsr(frs^{-1}(Ri)) \\ fgs(fgs^{-1}(Gi)) \\ fbs(fbs^{-1}(Bi)) \end{pmatrix} = T \cdot \begin{pmatrix} Ri \\ Gi \\ Bi \end{pmatrix} \qquad (7)$$

Experimental Results

To test the performance of the proposed approach, a real-world example was chosen. The test case is a conversion Camera RAW → sRGB. Camera RAW is a preferred format for most high-end photographers. It has certainly some advantages to other formats, but the apparent gamma of this space is about 1.0 and this makes difficult to convert it to a more perceptually uniform space like that one offered by sRGB, which has an apparent gamma of 2.2. The mismatch between gamma values makes the conversion hard when only a 3D LUT is available. Without linearization, 33 or more points per dimension would be needed in order to avoid artifacts.

Four test images were generated by the author using by 6 using, and 17 grid points, with and without linearization stage:

A. 6 points without linearization
B. 17 points without linearization
C. 6 points with linearization
D. 17 points with linearization

Images were stored and manipulated at 16 bits to avoid quantization effects. The color transform evaluated was setup using ICC profiles with far more precision that the test case. Final images were printed on a HP Design Jet Z3100, which is a large format printer aimed to photographer market. The 3DLUT was computed by using the LittleCMS color library.

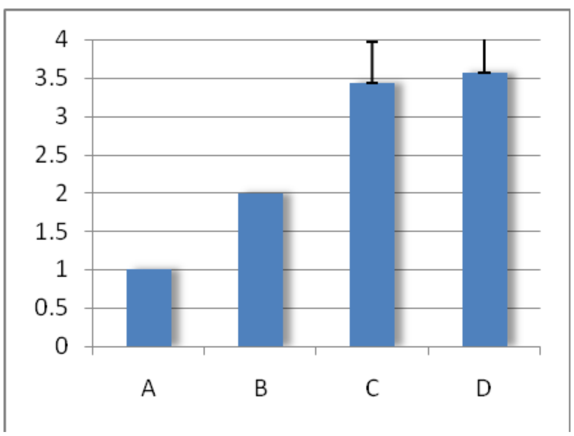


Figure 4, *Experimental results*

Five high-trained observers and two non-professionals were asked to evaluate the obtained images by sorting them by preference, putting the images they like more fist and those images with visible defects or with artifacts last. This procedure effectively rates each image in a scale of 1-4. Average score for each image is shown in a bar diagram. The Standard deviation is also depicted as error bars.

All observers agreed 6 point LUT without linearization performed worst, followed by 17 points without linearization. It is also remarkable to note that many observers perceived both plots using linearization C) and D) as identical.

Conclusions

In this paper we have presented a method to compute a linearization stage that may increase accuracy on lookup based RGB to RGB color transforms on both 8 and 16 bits. This method is especially effective when apparent gamma of input and output RGB spaces are quite different. A psychophysical testing involving both trained and untrained observers have confirmed it greatly improves the accuracy of transforms on real images.

References

[1] Kang., H. R. Color Technology for Electronic Imaging Devices. s.l. : SPIE Optical Engineering Press., 1997.
[2] Color gamut mapping and the printing of digital color images. Stone, Maureen C., Cowan, William B. and Beatty, John C. 1988, ACM Transactions on Graphics (TOG), pp. Vol. 7, No. 4, pp 249-292.
[3] Methods for characterizing CRT displays. Berns, R.S. Volume 16, Issue 4, May 1996, Displays, pp. Pages 173-182 .
[4] LCD versus CRTs color calibration and gamut considerations. Sharma, G. April 2002. proceeding of the IEEE. pp. vol 90, no. 4, pp. 605-622.
[5] Eilers, P.H.C. Smoothing and interpolation with finite differences. [book auth.] P.S. Heckbert. Graphic Gems IV. s.l. : Academic press., 1994.
[6] Johnson, Roger A. Horn and Charles R. Matrix Analysis. Section 7.2. s.l. : Cambridge University Press, 1985.
[7] Vrhel, M.J. Trussell, H.J. Color device calibration: a mathematical formulation. Springboro, OH : Color Savvy Syst. Ltd, 1999.
[8] Fairchild, MD. Color Appearance Models, 2nd ed. s.l. : John Wiley & Sons, 2005.
[9] CIE. A color appearance model for color management systems: CIECAM02. 2003.

Author Biography

Marti Maria is a color engineer in the imaging and color group at the large format printer division of Hewlett-Packard. He has a degree in telecommunication engineering from the Universitat Politécnica de Barcelona. Marti is also the author of well-known open source color-oriented packages like the LittleCMS CMM and the LPROF profiler construction set.