Journal of Imaging Science and Technology® 54(3): 030402–030402-14, 2010.
© Society for Imaging Science and Technology 2010

Color Management Using Optimal Three-Dimensional Look-Up Tables

Satyam Srivastava and Edward J. Delp[▲]
Video and Image Processing Laboratory (VIPER), School of Electrical and Computer Engineering, Purdue University, West Lafayette, Indiana
E-mail: ace@ecn.purdue.edu

Thanh H. Ha and Jan P. Allebach[▲]
Electronic Imaging Systems Laboratory (EISL), School of Electrical and Computer Engineering, Purdue University, West Lafayette, Indiana

Abstract. *Digital imaging has seen an unprecedented growth in the past five years. The variety of imaging systems available to users to create and view visual data is enormous. Color management has become an important aspect of modern imaging and display systems. Color profiles have been the de facto tool for achieving faithful visual reproduction for a long time. In this article, the authors discuss issues associated with profile-based color management systems. The authors describe an alternative approach motivated by the problem of visually matching two known display devices. The authors use a model-based method to achieve this goal and propose realizing the method with simple table look-up operations. The authors devise a framework for designing look-up tables (LUTs) which are optimal in terms of color reproduction on the displays based on resource constraints. The LUT-based color management system is shown to be more accurate and memory-efficient than a comparable International Color Consortium profile-based system.* © *2010 Society for Imaging Science and Technology.*
[DOI: 10.2352/J.ImagingSci.Technol.2010.54.3.030402]

INTRODUCTION

The quantity of visual data continues to grow as more users, both amateur and professional, create images using a wide range of capture devices. Digital images and video have become immensely popular with the decrease in costs of cameras, displays, and software tools. Similarly, with the ease of delivery, storage, and display, viewers have enormous choice of display devices on which to view content. A digital image is a representation of two-dimensional visual data and digital video can be considered a time-indexed sequence of digital images. A display device recreates a likeness of the visual data using the digital representation. These devices may use different technologies [such as cathode-ray tube (CRT), liquid crystal display (LCD), and projection systems], and hence, in most cases it is unlikely that visual data viewed using these devices would appear the same. More specifically, a digital image or video when viewed on such displays will not produce the same visual stimulus (measured objectively or subjectively). For many applications the output visual stimulus is the most important result. A typical scenario of visual data delivery is illustrated in Figure 1. The content creator (photographer) would like to ensure that each user obtains the "correct" output independent of their display device.

Let the set $\mathfrak{R}$ consist of all still and moving visual data and the set $\mathfrak{I}$ contain all digital image and video data. Then, using the notation shown in Fig. 1, $e_{\text{real}}, e_{\text{photograher}}, e_1, e_2, \ldots, e_n \in \mathfrak{R}$ and $i, j \in \mathfrak{I}$. The goal is to achieve $e_1 = e_2 = \cdots = e_n = e_{\text{photographer}}$. That is, every display should look the way the content creator desires it to look. It should be noted that our reference for visual matching is the visual output as "desired" by the content creator. This is important because a digital image (or video) often undergoes a series of enhancements/processing before being delivered to a user and hence, may not resemble e_{real}. Finally, $\Psi : \mathfrak{I} \to \mathfrak{R}$ represents the basic functionality of how a display device maps its input to its output.

Color management is the general collection of techniques used for achieving a "visual match" among display/printing devices. It is an important part of digital imaging systems. Personal computers and workstations provide support for color management at multiple levels including the operating system and application levels.[1,2] Most color management systems are based on sharing device properties using *color profiles*. Color profiles provide a way of transforming an image from the color space of a device to an independent color space which is known as the profile con-

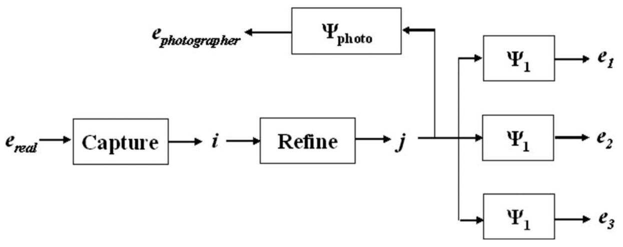

Figure 1. Typical visual data delivery/display framework.

[▲]IS&T Member.

Received Sep. 16, 2009; accepted for publication Mar. 22, 2010; published online May 7, 2010.
1062-3701/2010/54(3)/030402/14/$20.00.

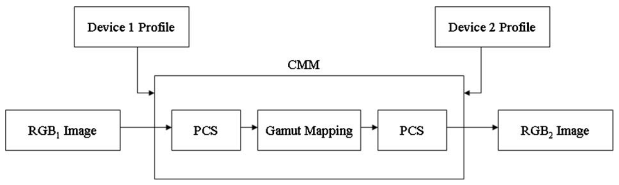

Figure 2. A profile-based color management system.

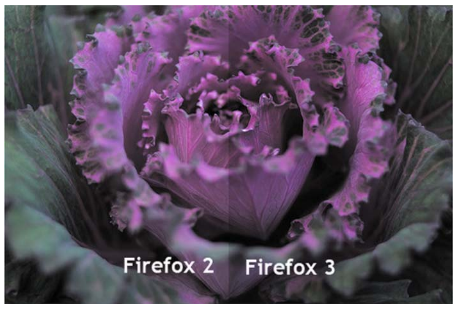

Figure 3. An illustration of color profile support in web browsers (courtesy: Deb Richardson[6]).

nection space (PCS). An example of a profile-based color management system (CMS) is shown in Figure 2. The various transformations between the device spaces and the PCS are performed in a color management module which is also responsible for gamut mapping that may be required, and is part of the user's display system. These operations may be implemented in hardware or software (or a combination of both) depending the display system and the application.

Although the profile-based approach to color management is widely used, there are aspects which limit its use for modern display systems and user requirements. Many of these issues have been reported in the literature[3,4] and are summarized here:

- Color management is usually believed to be a tool used by professionals so that most casual users do not calibrate their display devices for extended periods of time.[5] While generating a color profile does require some skill, in many cases, even the standard profiles are not used. This effectively "turns off" color management.
- For systems with color management and a calibrated display, the profile-based system will work only if the applications used support color profiles. The internet has become a major vehicle for delivering digital images and the lack of support for color profiles in many web browsers is disappointing.[6] FIREFOX recently introduced profile support in its 3.* releases. Figure 3 shows an illustration wherein a digital image with an embedded color profile is viewed using two versions of FIREFOX (the images were screen-captured and pasted together for comparison).
- While the commonly used profile format is well-standardized by the International Color Consortium (ICC),[7] some vendors have designed their own proprietary formats which are not compatible with the ICC format. The Windows Color System (WCS) introduced with WINDOWS VISTA™ is one such example.[8] This raises the issue of cross-platform operation because the color management system cannot use a profile created in a noncompliant format.
- As illustrated in Fig. 2, a profile-based color management system consists of multiple color space transformations. This requires both rigorous mathematics and implementation of complex algorithms (such as gamut mapping, white balancing, and chromatic adaptation). Clearly, this approach may not be suitable for low power mobile display systems. This is important because most mobile platforms (which include iPods™, mobile phones, and personal digital assistance devices) have limited computing power but are highly popular for viewing images.
- This approach is useful for digital images but is not used with digital videos. In fact, color management for consumer video is a fairly new topic.[9] An important application dealing with the problem of color management is the cinema industry.

We approach the problem of color management by examining the case of motion pictures. More specifically, we consider features compatible with the Digital Cinema Initiatives specification[10] with the goal of achieving $e_{\text{user}}=e_{\text{artist}}$, where e_{artist} is the visual output obtained during cinema *post production*.[11] During post production, the motion picture undergoes several transformations aimed at improving the audio-visual experience as well as operations such as editing and addition of special effects. To justify the efforts of artists in obtaining the "correct" visual experience for the content, it is important that the visual output obtained when the motion picture is screened in a cinema theater be as close as possible to that observed by the artists during the post production process.

From a color management point of view, this problem becomes one of visually matching two display devices—the cinema theater screen and the displays used in the post production operations. The traditional solution for the cinema industry has been the use of custom displays for the post production. These displays closely match the visual attributes of the cinema theater and help the artists preview a rendering of the motion picture, as it would be seen in the cinema theater. The use of custom displays is expensive and not very flexible. We also note that a profile based color management may not be applicable in this situation. This is primarily because the hardware in a cinema theater projection system usually does not have the capability to utilize profiles for color correcting a full length motion picture.

In this article we propose a new approach to color management which is both accurate and hardware-friendly. Our solution is based on the assumption that a reference display device exists, and all other display devices can be matched to this reference. This consists of two steps. In the first step, we develop a method for perceptually matching a given display with a known reference display. Perceptual similarity[7,12] is said to be achieved when two colors match in the CIE LAB

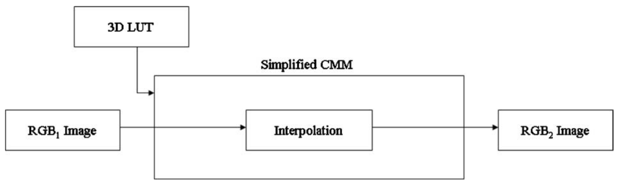

Figure 4. Our proposed LUT-based color management system.

space after accounting for chromatic adaptation of the human visual system[13] and ensuring a common white point. We select perceptual matching because it is more suitable for reproduction of motion content on systems with high dynamic ranges and under different viewing conditions.[7] This first step may be completed offline. In the second step, this method is realized as a faster and simpler operation, by using a look-up table (LUT). This is illustrated in Figure 4. We will show that this method provides better color reproduction because we develop a specific solution to match two display devices unlike matching a display to an unknown device using its profile.

It should be noted that the advantages of generating a direct device-to-device transformation, as we described above, have been recognized by color scientists. The ICC[7] defines a type of profile which contains the information about the transformation between two known devices. Such profiles are known as *device-link profiles*. While a color management system utilizing device-link profiles promises better color accuracy and lower computational complexity, these profiles have been largely used in printing systems but for a different reason. In high quality print systems, the transformation from one device's CMYK color space to another device's CMYK space may suffer loss of information when an intermediate space, the profile connection space, or the PCS (for example, the CIE LAB space) has only three data channels. In such cases, device-link profiles can be used to bypass the PCS and can also be used to preserve the neutral colors[14,15] by preventing CMY components from leaking-in when only the K component may be needed.

These profiles can be created using the colorimetric data measured directly from the two devices. However, many profiling tools can generate a device-link profile indirectly, using two ICC device profiles.[15–17] Balonen-Rosen and Thornton[18] described a method which allows a user to interactively modify the existing information in color profiles (including device-link profiles) to improve the quality of color management. This local color correction technique is useful when an image rendered on a destination device contains multiple color regions that are unacceptably different from the image rendered on a source device.

Our LUT-based approach is different from a system that uses device-link profiles in the following ways. The ICC specification does not allow a device-to-device transformation (contained in a device-link profile) to be represented only by a look-up table. The four acceptable algorithmic models for such transformations shown in Figure 5, on page xii of the ICC standard[7] consist of one or more "processing elements," including a multidimensional LUT. The two configurations which utilize a multidimensional LUT also include at least two other processing elements. Thus, a hypothetical ICC-compatible CMS that processes only device-link profiles will be more complex than our proposed CMS.

Second, our approach allows greater flexibility in the construction (and usage) of the LUT. As we describe later in this article, we generate a LUT by selecting a LUT size and interpolation method and an optimal nonuniform set of sample points. The ICC standard only allows interpolation on a uniformly sampled data set. Our method allows trade-offs between accuracy, complexity, and required memory.

Thus, our end-to-end design can be more accurate and require fewer resources than a profile-based approach. Since our proposed CMS requires only table look-ups and interpolations, it is suitable for hardware and embedded implementations. For example, one could design a display device with an embedded color management capability because of the simplicity of our proposed system.

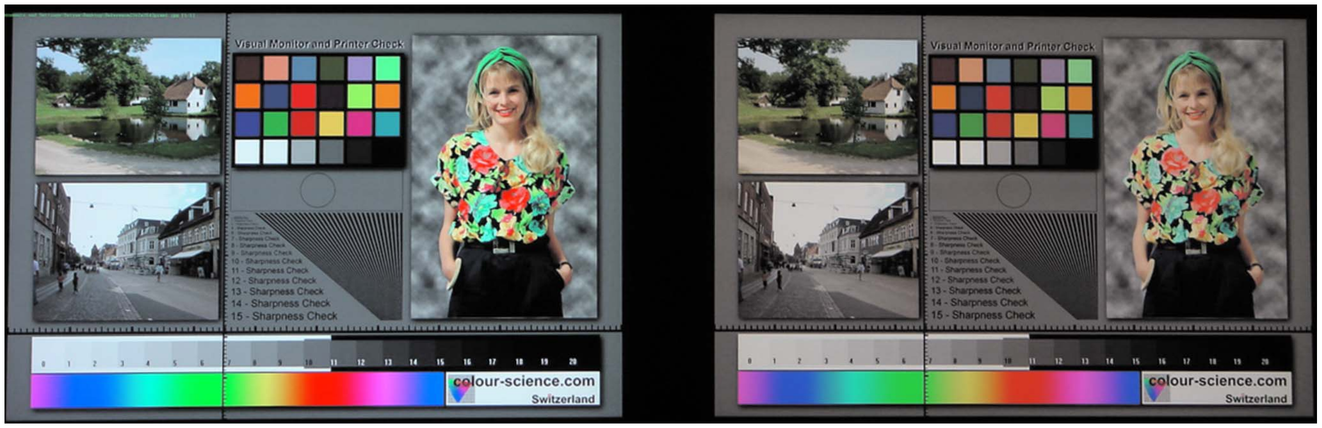
Visual Monitor and Printer Check
colour-science.com
Switzerland
Visual Monitor and Printer Check
colour-science.com
Switzerland

Figure 5. A test image displayed in a dark room using two monitors that are different models from the same manufacturer (test image—courtesy: Color Science AG, Switzerland[19]).

The rest of this article is organized as follows: in the section Analytical Transformations Using Device Models we briefly review some of the methods proposed for visual matching of two displays. In particular we describe our method for achieving perceptual similarity between two display devices and represent it as a model of the color management problem. This model is approximated by three-dimensional (3D) LUTs in the section Efficient Realization Using Look-Up Tables. We also describe a process for optimizing the parameters of a LUT such that it results in the lowest approximation errors given resource constraints. In the Experimental Results section, the performance of our LUT-based color management system is evaluated. We devise experiments to determine the accuracy of the color reproduction and compare it with that of an ICC color profile-based system.

ANALYTICAL TRANSFORMATIONS USING DEVICE MODELS

Consider a situation where we have two arbitrary display devices. One of the displays is used to enhance, process, and distribute digital images. This display is denoted as the *reference device*. The second display is representative of the general class (or model) of devices used to view the images and is referred to as the *user device*. It is natural that the visual output of the two devices will be different unless some form of color management is used. Figure 5 shows an example where a test image is displayed on two Dell LCD monitors belonging to different generations and under factory default settings. The images can look very much different.

In many application scenarios a user display may not have a color management system and hence embedding an ICC profile in the image may not be useful. An alternative method is to change the digital representation of the images before providing them as inputs to the *user* monitor such that the visual outputs are matched. In other words, one needs to design a function $f:\mathfrak{I}\rightarrow\mathfrak{I}$ such that if $i\in\mathfrak{I}$ is the input to the reference device, $f(i)\in\mathfrak{I}$ is the input to the user device that ensures $\Psi_{\mathrm{ref}}(i)\approx\Psi_{\mathrm{usr}}(f(i))$. In our discussions below, the device space $\mathfrak{I}$ is the RGB color space with 8 bits used to encode the digital input for each color channel and $\approx$ represents perceptual similarity as defined in the Introduction. That is, perceptual similarity[7,12] is said to be achieved when two colors match in the CIE LAB space after accounting for chromatic adaptation of the human visual system[13] and ensuring a common white point.

The problem of visual matching of two display/print media has been studied and many schemes have been proposed. These methods in turn rely on mathematical modeling of display devices and the transformations between different color spaces. Characterization techniques for both CRT and LCD displays have been devised.[20–22] Similarly, methods for monitor-printer matching, gamut mapping, and evaluation of color reproduction quality have been proposed.[23–27]

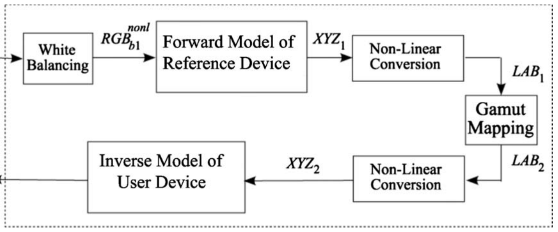

Figure 6. Block diagram of the model nonlinear transformation with white point correction (NLXWP).

We use the method we previously described[28] to obtain the function f. This method uses models of the display devices using CIE LAB as the intermediate space between the device native color spaces. A block diagram of the method is shown in Figure 6. We refer to this method as *nonlinear transformation with white point correction* (NLXWP). This method is used because it improved upon the commonly used display device model, which consists of gray balancing curves and a matrix transformation, without making the device model difficult to invert.

In Fig. 6, the block labeled "forward model of reference device" represents the function $\Psi_{\mathrm{ref}}:\mathrm{RGB}\rightarrow XYZ$. This model consists of gray balancing and a linear transformation, described next. Similarly, the "inverse model of user device" represents a function $\Psi^{-1}_{\mathrm{usr}}:XYZ\rightarrow\mathrm{RGB}$ and consists of operations that are the inverse of the forward model. Thus, NLXWP achieves an overall transformation $f:\mathrm{RGB}\rightarrow\mathrm{RGB}$ by coupling the two device models.

We next describe some of the salient features of NLXWP (refer to our previous publication[28] for a complete description):

- Gray balancing is used to convert between nonlinear and linear RGB spaces and is achieved using one-dimensional (1D) look-up tables, one for each color channel. In this article, we explicitly use the term *nonlinear* RGB to refer to "gamma corrected"[29] RGB as opposed to linear RGB values which are proportional to the photon count.
- A single 3×3 transformation matrix is used as the transform between the linear RGB space and the CIE *XYZ* space. This matrix is obtained by optimizing the conversions with respect to training data. All the colorimetric measurements were done using a Photo Research PR-705 spectroradiometer to record the CIE *XYZ* values from the display devices.
- The method incorporates chromatic adaptation capability of the human visual system to achieve a perceptual match between the two displays. Based on the von-Kries model,[13,30] a chromatic adaptation transformation from XYZ_1 in the viewing condition of the reference display (denoted as display 1) to XYZ_2 in the viewing condition of the user display (denoted as display 2) is formulated as

$$\begin{bmatrix} X_2 \\ Y_2 \\ Z_2 \end{bmatrix} = \boldsymbol{H}^{-1} \begin{bmatrix} \frac{L_{w_2}}{L_{w_1}} & 0 & 0 \\ 0 & \frac{M_{w_2}}{M_{w_1}} & 0 \\ 0 & 0 & \frac{S_{w_2}}{S_{w_1}} \end{bmatrix} \boldsymbol{H} \begin{bmatrix} X_1 \\ Y_1 \\ Z_1 \end{bmatrix}, \tag{1}$$

where LMS_{w_i}, $(i=1,2)$ are the LMS cone responses[30] to the reference white w_i of Display i and $\boldsymbol{H}$ is a nonsingular matrix which transforms XYZ to LMS. In the NLXWP model, we assume the reference white XYZ_{w_i}, $(i=1,2)$ of the two displays share the same chromaticity coordinates. Under these conditions, it can be shown that Eq. (1) simplifies to

$$\mathrm{LAB}_2 = \mathrm{LAB}_1, \tag{2}$$

where LAB_i, $(i=1,2)$ is the LAB 3-tuple obtained by converting XYZ_i in Eq. (1) to the LAB space with respect to the reference XYZ_{w_i}.

- It also includes a "white-balancing" step to ensure a common white point for the two displays.[31] The balancing is implemented in the linear RGB space as follows:

$$\begin{bmatrix} R_b^{\mathrm{lin}} \\ G_b^{\mathrm{lin}} \\ B_b^{\mathrm{lin}} \end{bmatrix} = \begin{bmatrix} \frac{R_{dw}^{\mathrm{lin}}}{R_{nw}^{\mathrm{lin}}} & 0 & 0 \\ 0 & \frac{G_{dw}^{\mathrm{lin}}}{G_{nw}^{\mathrm{lin}}} & 0 \\ 0 & 0 & \frac{B_{dw}^{\mathrm{lin}}}{B_{nw}^{\mathrm{lin}}} \end{bmatrix} \begin{bmatrix} R_n^{\mathrm{lin}} \\ G_n^{\mathrm{lin}} \\ B_n^{\mathrm{lin}} \end{bmatrix}, \tag{3}$$

  where "b" and "n" denote "white-balanced" and "native," respectively, and "dw" and "nw" denote "desired white" and "native white," respectively.
- Gamut mapping is done using the straight chroma clipping (SCC) method which is simple and has been shown to work better than many other clipping and compression techniques.[25,32] Figure 7 shows the SCC method in which an out-of-gamut color is shifted to the destination gamut's boundary, while its lightness L^* and hue h^* values are preserved.

We evaluated the accuracy of NLXWP and computed the error statistics.[28] The transformations achieve an average error of 1.94ΔE units with a standard deviation of 0.80ΔE units. The testing set contained 625 RGB values in the form of a $5\times5\times5$ grid, whose elements were randomly selected. We later proposed further improvement in the display device models by the use of nonsquare transformation matrices (for example, 3×4 and 3×11), as well as multiple matrices for different color classes.[33] Such device models using multiple rectangular matrices are highly accurate with average errors around 0.5ΔE units and have been used in this article to simulate ideal display devices. The model NLXWP is developed for perceptually matching these two simulated devices and serves as the function f defined above.

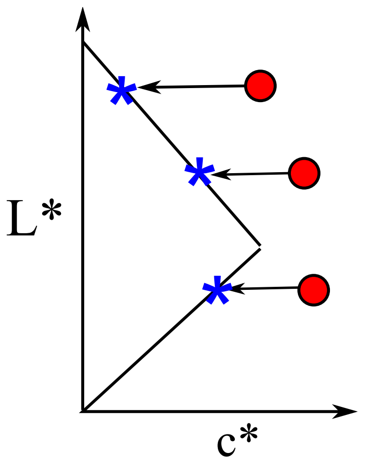

Figure 7. Illustration of straight chroma clipping (SCC): the red circles represent out-of-gamut colors and the blue stars represent clipped colors.

EFFICIENT REALIZATION USING LOOK-UP TABLES

The NLXWP function is computationally complex and may not be easy to implement for all displays. Therefore, we use 3D LUTs to realize the function in a computationally efficient manner. LUTs are a convenient tool used in many applications in image processing. Consider a function $y=f(x)$ which maps an input $x\in D$ to an output $y\in R$, where D and R are the domain and the range of f, respectively. The concept of a table look-up involves computing discrete values of f for a set of discrete inputs $S_D\subset D$ and storing them as a set $S_R\subset R$. In order to evaluate the function at a given input $x\in D$, we locate points in S_D which surround x. In other words, we identify the elements of S_D which form a box (or a polytope) such that x is an internal point of the box. These points are known as neighbors of x. Finally, $f(x)$ is obtained by interpolating between the known values of the function at its neighbors. The elements of the subset S_D for which the function is evaluated are known as the *sample* points or *knot* points.

Look-up tables have been used to realize color space transformations between different media.[34] Some of these methods were in use well before the development of standardized profile-based methods. In 1975, Pugsley[35] described a method for color correcting the output of a scanner using pre-computed values stored in the memory. In 1978, Kotera described LUT-based techniques for Bayesian color separation.[36] Researchers at Polaroid Corporation developed a LUT-based method in 1989 for managing color transformations between photographic films and printed media.[37] McCann describes 3D LUT-based methods to perform color space transformations for various applications, including matching of photographic films with printed media, and transforming between the CIE LAB space and a more uniform Munsell color space.[38,39] These transformations were designed on a $8\times8\times8$ hardware LUT and achieved high

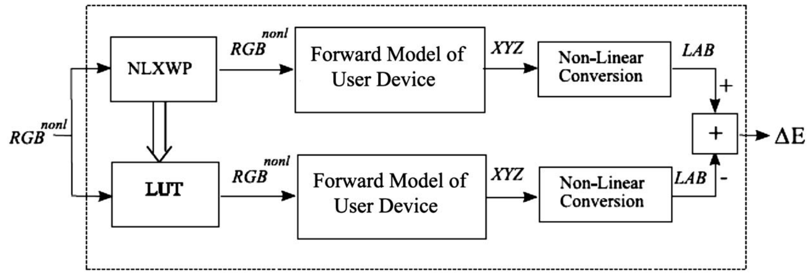

Figure 8. Block diagram of the system for constructing and evaluating a 3D LUT (LUTEVAL).

levels of accuracy. The LUT-based techniques were also used for color correction in the successful efforts to create life-size replicas of fine art.[40]

We previously described the applicability of a 3D LUT for achieving complex transformations.[41] We also described a method for obtaining the optimal parameters for such a system. In the following sections, we provide more detail about the approach and some motivation behind the proposed scheme for the optimal sampling of the RGB space. The desired LUT is constructed using the NLXWP function and also cross-validated against it. Figure 8 illustrates the method of generating and evaluating the LUT using the NLXWP model. This is referred to as LUTEVAL and is used (as a black box) in our optimization setup. The block arrow between NLXWP and LUT represents the offline process of generating the table by evaluating NLXWP at the desired points. The LUT generated by LUTEVAL is stored as a file and, hence, this step does not occur when the LUT is evaluated against the model.

Let $f(\,)$ represent the nonlinear transformation (NLXWP) using the device models. Since our transformations map nonlinear RGB space to nonlinear RGB space, let $[r,g,b]^{\text{in}}$ and $[r,g,b]^{\text{out}}$ indicate the input and output vectors, respectively. Next, we let the table look-up be described by a function $\Phi(\,)$. As in the previous sections, a subscript 1 stands for the first monitor (reference/target device) and subscript 2 stands for the user device or the second monitor. Then, for the same input $[r,g,b]^{\text{in}}$ to the two functions:

$$[r,g,b]^{\text{out}}_{\text{LUT}} = \Phi([r,g,b]^{\text{in}}) \tag{4}$$

and

$$[r,g,b]^{\text{out}}_{NL} = f([r,g,b]^{\text{in}}). \tag{5}$$

Using the forward monitor models, we can obtain the corresponding values in the LAB space and compute the difference between the colors:

$$\Delta E = \|[\text{LAB}]^{\text{out}}_{NL} - [\text{LAB}]^{\text{out}}_{\text{LUT}}\|. \tag{6}$$

This process is illustrated in Fig. 8.

The transformation f requires a 3D LUT with a vector $[r,g,b]$ as input. Each entry in the LUT is also a vector $[r,g,b]$ corresponding to the color-corrected digital input for the user device. For 8-bits per channel, the input space is of size $256\times256\times256$ but practical LUTs would be much smaller. Consider a LUT of size $n_R\times n_G\times n_B$ and sample points selected according to the set

$$\Omega = \{r_0,r_1,\ldots,r_{n_R-1}\}\times\{g_0,g_1,\ldots,g_{n_G-1}\} \\ \times\{b_0,b_1,\ldots,b_{n_B-1}\}, \tag{7}$$

where $\times$ denotes a Cartesian product. Then, the function $\Phi(\,)$ can be specified as:

$$\Phi([r,g,b]) = \begin{cases} f([r,g,b]), & \forall[r,g,b]\in\Omega, \\ £([r,g,b]), & \text{otherwise}. \end{cases} \tag{8}$$

Here $£(\,)$ represents an interpolation operation which is used to transform the input values not aligned with any of the table entries.

The accuracy of a LUT-based transformation is measured in terms of the deviation of the colors in ΔE units. Let S_T be a training set of N RGB values taken randomly from the RGB space. The difference between the colors produced by the NLXWP model (f) and the LUT (Φ) is given by

$$\Delta E_i = \|[L,A,B]^{\text{out},i}_{NL} - [L,A,B]^{\text{out},i}_{\text{LUT}}\|. \tag{9}$$

An average of all the differences is referred to as the transformation error:

$$\overline{\Delta E} = \frac{1}{N}\sum_{i=0}^{N-1}\Delta E_i. \tag{10}$$

This average error over the selected training set serves as our yardstick for evaluating the effects of different parameters of a LUT-based system.

Having constructed a method for using a function to generate and evaluate a LUT, we next focus on the various factors which affect the accuracy with which the LUT can approximate the function. Given resource constraints, these can be broadly categorized as follows.

The size of each dimension of a multidimensional LUT is related to the relative importance of each dimension. In our problem, this relates to how many sample points are selected from each of R, G, and B color channels (n_R,n_G,n_B, respectively) such that the overall LUT complies with the overall size constraint. As an example, one can construct a $10\times10\times10$ LUT or a $10\times12\times8$ LUT if no more than

1000 table entries are allowed. In our experiments, we assume an equal number of sample points are collected from each color channel. Such LUTs are referred to as *regular*.

A typical (except boundary) point in the 3D space has eight nearest neighbors where the value of the function is known. An interpolation algorithm may use all the eight values or a subset of these to determine the function's value at the unknown point. It may also assign unequal weights to the known values according to the neighbor's "distance" from the point. Thus, the method of interpolation is an important aspect that affects the accuracy of a LUT system.

The sampling set Ω contains the elements of the discrete domain of the function. This set is representative of the function and, hence, the choice of its elements plays a key role in the successful interpolation of other values. We also define a special type of sampling for a regular LUT in which $r_i = g_i = b_i \;\; \forall \;\; i \in \{0, 1, \ldots, n_R - 1\}$. Such tables will be called *symmetric* and regular tables not meeting this condition will be known as *nonsymmetric*.

Interpolation

Interpolation has been well explored and several interesting methods have been proposed in one and more dimensional spaces and on regular and irregular shaped data grids. Bala and Klassen[42] provide a survey of the commonly used interpolation methods, in the context of color transformations. They also describe some acceleration techniques that can result in faster operation under suitable conditions. Kidner et al.[43] reviewed many interpolation and extrapolation methods as applicable to the problem of building sophisticated digital elevation models used (among others) in terrain analysis, infrastructure design, and global positioning. In the following discussion, we describe a simple interpolation scheme in three dimensions. We also present some results demonstrating the relative effectiveness of two derived schemes in our problem of approximating the NLXWP model. Note that this discussion is included primarily for the purpose of completeness and as an illustration of the effect of choosing an interpolation technique on the performance. The focus of our optimization efforts is on the sampling of the RGB space.

In the most generalized situation, an input point $[x,y,z]_s$ (whose output needs to be predicted) can have k neighbors $[x,y,z]_i$ for $i = 0, 1, \ldots k-1$. Let $d(i,s)$ be some metric of distance between the points i and s. Note that each neighbor is an entry of the table and hence their outputs $f([x,y,z])_i$ are known. Then using interpolation,

$$f([x,y,z])_s = \psi(f_i, d_i) \quad \text{for} \quad i = 0, 1, \ldots, k-1, \tag{11}$$

where ψ is determined by the method used.

For example, a simple linear interpolation in one-dimensional (1D) space would be

$$f_s = [f_0 d(s,1) + f_1 d(s,0)]/[d(s,0) + d(s,1)], \tag{12}$$

where $d(s,i)$ is the Euclidean distance between s and its ith neighbor. A simple extension to three dimensions (3D) would be one where f_s is evaluated as a weighted sum of f_i

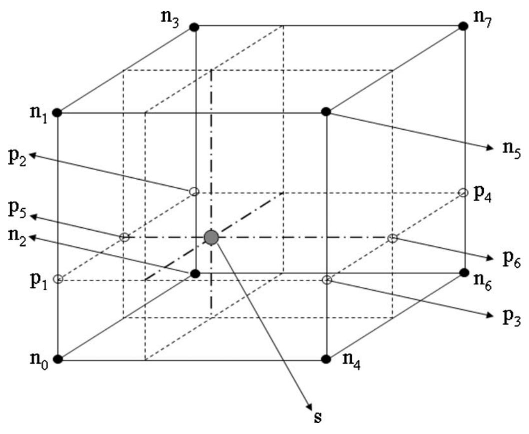

Figure 9. An illustration of the neighbors of a point in 3D space.

for $i = 0, 1, \ldots, 7$, with each f_i weighted by the Euclidean distance of the other neighbor (and normalized by the sum of these distances) from the required point s as shown in Figure 9 where the neighbors are denoted by n_i. Note that in our problem, the 3D box formed by the eight neighbors is not required to be a cube, or even a regular polytope because of nonuniform sampling. A number of schemes exist for interpolation using different neighbor sets and different distance metrics. Shepard's[44] interpolation is a general framework for interpolation on an irregularly spaced data. Another general technique known as sequential linear interpolation optimizes an nD interpolation problem by sequentially reducing the dimensionality while maintaining optimal performance.[45]

As noted earlier, a point in a 3D space has eight nearest neighbors located at the vertices of a bounding box. It is possible to use a subset of these eight points for interpolation, such as in tetrahedral interpolation.[46] In our experiments for color space transformation, we employ interpolation methods which use all the eight neighbors of the unknown point. Let $[r,g,b]_s$ be the input value for which an output is needed. We define $[r,g,b]_i$ as the eight neighbors of the input, where $i = 0, 1, \ldots, 7$, and the order/position is as indicated in Fig. 9. Observe that $[r,g,b]_0 = [\lfloor r \rfloor, \lfloor g \rfloor, \lfloor b \rfloor]_s$ where $\lfloor \; \rfloor$ denotes the nearest lower neighbor, that is, the nearest neighbor of s which can be reached by only reducing the value of the color channel. Note that in this context $\lfloor \; \rfloor$ does not represent the usual integer flooring. Similarly, $[r,g,b]_7$ is the nearest upper neighbor.

The interpolated value r^{out} can now be obtained in the following manner. In the following equations, the symbols p_i represent intermediate points as shown in Fig. 9 and t_i are the corresponding function values. The symbol r_i represents the first component of $[r,g,b]_i$ and l_i is a metric of distance of the ith neighbor from s. The values of g^{out} and b^{out} can be similarly obtained;

$$t_1 = (r_0 l_1 + r_1 l_0)/(l_1 + l_0), \tag{13a}$$

$$t_2 = (r_2 l_3 + r_3 l_2)/(l_3 + l_2), \tag{13b}$$

$$t_3 = (r_4 l_5 + r_5 l_4)/(l_5 + l_4), \tag{13c}$$

$$t_4 = (r_6 l_7 + r_7 l_6)/(l_7 + l_6), \tag{13d}$$

$$t_5 = (t_1 l_{p_2} + t_2 l_{p_1})/(l_{p_2} + l_{p_1}), \tag{13e}$$

$$t_6 = (t_3 l_{p_4} + t_4 l_{p_3})/(l_{p_4} + l_{p_3}), \tag{13f}$$

and finally,

$$r^{\text{out}} = (t_5 l_{p_6} + t_6 l_{p_5})/(l_{p_6} + l_{p_5}). \tag{14}$$

In the above formulation, we can define $l_i = d(s,i)$. Then, the known value at a neighbor is linearly weighted by the Euclidean distance of the opposite neighbor from the unknown point. This can also be viewed as weighting the value at a neighbor with its inverse distance from the unknown point. For example, Eq. (13a) can be written as

$$t_1 = (r_0 l_0^{-1} + r_1 l_1^{-1})/(l_0^{-1} + l_1^{-1}). \tag{15}$$

Note that this equivalence holds only when interpolating between two points. We refer to this scheme as *separable inverse distance interpolation*. We can also define $l_i = d^2(s,i)$. The resultant scheme is referred to as *separable squared inverse distance interpolation* in this article.

The choice of nonlinear weighting of the function values becomes particularly significant because we are selecting the grid points for the LUT in a nonuniform fashion, which we describe in the following section. Shepard presented a method for interpolating on an irregular grid in a two-dimensional space.[44] The central idea of the method can be described by

$$f_s = \begin{cases} f_i & \text{if } d(s,i) = 0 \text{ for any } i \in \{0, P-1\}, \\ \dfrac{\sum_{i=0}^{P-1} f_i d^{-u}(s,i)}{\sum_{i=0}^{P-1} d^{-u}(s,i)}, & \text{otherwise,} \end{cases} \tag{16}$$

where P specifies the total number of neighbors used in the interpolation and the parameter u determines the weighting of grid points. Shepard goes on to describe the criteria for the selection of u. Based on empirical results, Shepard states that a value of $u=2$ would provide satisfactory results for surface mapping and is computationally inexpensive (compared to fractional values).

Table I. Average errors (ΔE units) obtained when approximating the NLXWP model with uniform LUTs using two interpolation methods.

Size	$\overline{\Delta E}_{\text{inv}}$	$\overline{\Delta E}_{\text{sq,inv}}$	$\overline{\Delta E}_{\text{sq,inv}} - \overline{\Delta E}_{\text{inv}}$
$9\times9\times9$	5.49	3.79	1.70 (30.9%)
$15\times15\times15$	3.10	2.16	0.94 (30.3%)
$21\times21\times21$	2.30	1.65	0.65 (28.3%)

We note that by defining $l_i = d^u(s,i)$, we can approximate Shepard's interpolation by a sequence of pairwise interpolations represented by Eqs. (13) and (14). It was observed in our experiments that the use of squared inverse distance $[l_i = d^2(s,i)]$ results in lower errors when 3D LUTs were used to approximate NLXWP than when using simple inverse distance $[l_i = d(s,i)]$.[41] This is summarized in Table I where the average errors of the two schemes are presented with respect to a randomly generated testing set of 1000 RGB values. The errors were obtained using the LUTEVAL block described earlier. Here, $\overline{\Delta E}_{\text{inv}}$ is the error corresponding to the separable inverse distance scheme and $\overline{\Delta E}_{\text{sq,inv}}$ is the error corresponding to the separable squared inverse distance scheme.

All the LUTs were generated by uniformly sampling the RGB space in these experiments. This ensures that the difference in the errors resulted only from the choice of the interpolation techniques. Note that the impact of squared inverse distance interpolation decreases (in both absolute ΔE units and in percentage) as the size of the LUT increases. This is expected and emphasizes the role that proper selection of the system parameters can play when the resources are constrained. We also performed this experiment with the nonseparable Shepard's interpolation as given in Eq. (16), with $P=8$. We observed that the nonseparable method resulted in lower average errors compared to the two separable methods described above. However, the difference between the average errors becomes smaller when interpolating on lattices optimized with respect to the corresponding interploation schemes. It should be emphasized that the regular structure of the separable methods makes them more suitable for hardware applications particularly when interpolating in a 3D space. Therefore, in this article, we only provide detailed results based on the separable inverse distance interpolation and the separable squared inverse distance interpolation. In subsequent sections, the squared inverse distance interpolation is used because it produces lower errors.

We would also like to mention that the use of squared inverse distance interpolation has been selected on the basis of accuracy and the simplicity of design. We optimize the sampling of the RGB space (in the next section) with the assumption that this interpolation would be done when using the optimal LUTs. However, other interpolation schemes may be selected based on application-specific requirements such as simplicity, regularity, and accuracy. While an interpolation scheme based on only linear interpolations would be simple and involve lower computational costs, an interpolation method based on higher degree polynomials, or on splines, may result in a smoother interpolated surface.

Splines are commonly used in image interpolation[47,48] because they can produce high levels of accuracy and can be selected to satisfy regularity requirements. Splines can be represented as piecewise polynomial functions of a specified degree. Thus, unlike higher degree polynomials, splines can be designed to avoid excessive oscillation. The theory of basis splines (B-splines)[49,50] allows interpolation in such a way that each piecewise function is a linear combination of a set of basis functions. This results in a reduction in the complexity of spline interpolation. Two common methods for extending B-splines to a multidimensional space are the use of tensor product splines[51] and the use of thin plate splines.[52]

We do not use spline interpolation for two reasons. The spline basis functions must be computed directly using the positions of the knot points. It is not practical for hardware implementations to precompute these basis functions if the knot points are irregularly spaced. Second, the actual interpolation uses a linear combination of the basis functions and the coefficients used as weights must also be computed before interpolation. Therefore, an implementation of splines in a high dimensional irregular space would be more complex.

We summarize this discussion by stating that our optimization framework is agnostic to interpolation and a simple modification of the cost function is needed to reflect the change.

Sampling

Subsampling of the RGB space is a complex problem that has been discussed in the literature. Monga and Bala[53] proposed a strategy for the optimization of the knot points (or nodes) based on a 3D importance function. They construct a significance function using the input distribution and the curvature of the analytical function to be approximated. Instead of trying to determine an exact solution to the problem of optimal node selection, they devise an efficient algorithm to obtain an approximate optimal lattice, in an iterative manner. They later proposed a method to jointly optimize node selection and the output values stored at the nodes[54] in order to minimize the expected interpolation error. However, a general solution for approximating 3D functions is not available, particularly when the function cannot be rigorously defined (in order, for example, to compute the curvature). Intuition suggests that a discrete domain Ω for a LUT constructed by uniformly sampling the input space may not deliver the optimal performance. It is possible that using more samples from the dark regions of the color space for Ω may lower the average error. This is mainly because a small change in the RGB values in the low lightness regions causes a large perceptual change in the response of the human visual system.

We experimented with different approaches to arrive at a sampling strategy that would provide an improved performance relative to uniform sampling. These are described below.

Sampling in LAB Space

A straightforward way to achieve perceptual uniformity would be to use the color space that best models the human visual perception. The goal would be to divide the CIE LAB space into n equal subspaces. The boundaries of these subspaces could then be translated back to the RGB space using the available nonlinear monitor models and a LUT could be constructed from these values. While this approach is attractive, it is intractable because the LAB space is not a regular polyhedron like the RGB space. It is quite possible that the space for a given display device is shaped like a deformed sphere. Locating the exact bounding surfaces of the LAB space might require enormous computation and it is unlikely that such a space can be divided into regularly shaped subspaces.

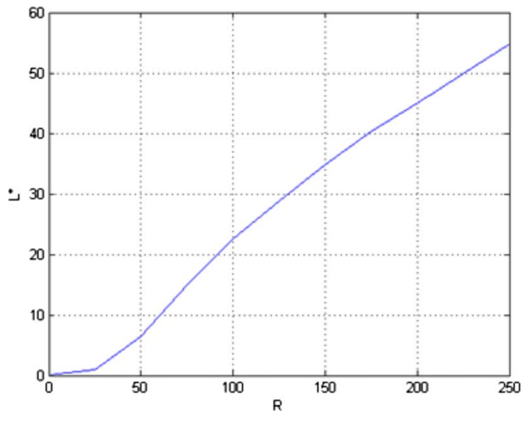

Figure 10. Variation in lightness (L^*) with digital red input R.

Uniform Lightness Sampling

It is tempting to try achieving at least uniform sampling in the CIE L^* dimension. One would expect that the function of lightness versus R (or G or B) would be concave with higher gradient near the dark region. Therefore, if the color channels are sampled for uniform L^*, it should result in more samples collected from the dark areas. A plot of nonlinear R versus L^* is shown in Figure 10. Although there is significant nonlinearity near the dark end, most of the function is nearly linear. However, this is not so unexpected if we understand the relationship between the native color space and the L^* axis. There are two basic steps in going from R to L^*—first R (along with G and B values) is converted to linear RGB by a power law correction (gray balancing), and then it is transformed to XYZ by a linear transformation. Finally, lightness is obtained using another power law. The two power law corrections reduce the effect of each other to a large extent and hence a near-linear behavior is observed for most inputs.

Optimal RGB Sampling

It is evident that no clear methodology seems to exist for achieving the best sampling map or the set Ω. Therefore, we try to obtain the optimal value of Ω using constrained optimization. Given a specification of the table dimensions $n_R \times n_G \times n_B$, the universe of all possible sampling maps (without any constraints) would be roughly $256^{n_R+n_G+n_B}$. Once we apply some obvious constraints (described below), this number reduces slightly but it is still $O(256^{n_R+n_G+n_B})$. Considering the fact that it is unlikely that a particular knot point is selected twice, a more accurate estimate of the size would be $\{256\times(256-1)\times\cdots[256-(n_R-1)]\}^3$. In simple terms, this is an optimization problem in a $n_R+n_G+n_B$ dimensional space.

An optimization problem is defined in terms of a cost function. In our case, we use the average prediction error $\overline{\Delta E}$

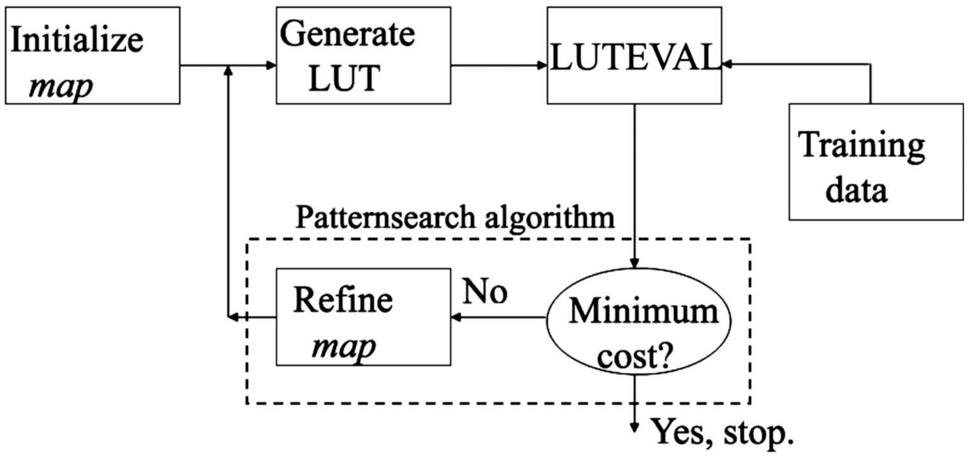

Figure 11. Block diagram of the optimization problem of subsampling the RGB space.

as the cost function given the training data set. We obtain a training data set S_T consisting of 1000 points (RGB triples) by constructing a $(10\times 10\times 10)$ grid in the RGB space:

$$S_T=\{r_0,\ldots,r_9\}\times\{g_0,\ldots,g_9\}\times\{b_0,\ldots,b_9\}, \tag{17}$$

where $\times$ denotes a Cartesian product. Initially all elements of this set were chosen randomly from the RGB space, but this approach often returns a skewed data set. Therefore, we enforce order by adding the eight corners of the RGB space to the set. The remaining 992 elements were randomly selected. The size of the training set is chosen to be significantly larger than the number of unknowns, thereby avoiding the possibilities of overfitting.

We proceed by inserting constraints which make the problem more tractable. First, we assume $n_R=n_G=n_B$ which means that we optimize only regular LUTs. Let

$$M_R=\{r_0,r_1,\ldots,r_{n_R-1}\}, \tag{18}$$

$$M_G=\{g_0,g_1,\ldots,g_{n_G-1}\}, \tag{19}$$

$$M_B=\{b_0,b_1,\ldots,b_{n_B-1}\}, \tag{20}$$

and

$$\text{map}=M_R\times M_G\times M_B. \tag{21}$$

Then

$$\Omega_{\text{opt}}=\arg\min_{\text{map}}\overline{\Delta E}, \tag{22}$$

where the vectors M_R,M_G,M_B are known as channel maps while map is the Cartesian product of the channel maps and contains all the knot points at which $\Phi(\,)$ is evaluated to construct the table. Ω_{opt} is the value of map such that the LUT based prediction model has the smallest $\overline{\Delta E}$ for a given training data set. The minimization is subject to the constraints that each element of map belongs to $\{0,1,\ldots,255\}$ and the elements of each channel map M_R,M_G,M_B are strictly increasing. For example, in the case of the red channel: $r_0<r_1<\cdots<r_{n_R-1}$.

The computation of $\overline{\Delta E}$ for a given map is a two step process consisting of generating the LUT by evaluating the function according to map and then using this LUT to transform elements of the training set S_T to obtain the average prediction error. Since this process cannot be rigorously defined, many optimization techniques (such as gradient-based schemes) cannot be used. Figure 11 illustrates the optimization problem graphically, and the problem is solved using the pattern search algorithm.

Pattern search[55,56] is a nongradient based optimization technique which can locate the global minimum of a given objective function with linear constraints. It has been used in a wide variety of optimization problems. At any iteration, the algorithm evaluates the objective function at every point in a set, known as a *pattern*. The pattern is expanded or shrunk depending on these values. More specifically, the pattern is expanded if any point in the set results in a lower cost than the current minimum; otherwise the set is shrunk (usually by a factor of two). Since the method does not require computation of derivatives to update the pattern, it can be used when the objective function is such that this information is not available or difficult to obtain. Hence, it is suitable for our optimization problem. The search for the optimum terminates when the pattern shrinks below a specified maximum size. The method has been shown to possess robust convergence properties.[57] Software implementations of the method can be obtained in the MATLAB Genetic Algortihms and Direct Search toolbox,[58] in the open source OPT++ library from Sandia Corporation[59] and from the authors of the algorithm.[60]

It should be noted that a symmetric sampling might allow greater control over the transformation of the neutral axis. This may be an important decision for certain applications that require pure neutral colors to be faithfully reproduced. Similarly, some applications (such as motion pictures) may place more importance on the skin tones. Our optimization setup provides an interesting method of specifying such preferences, which is by allowing the user to care-

Table II. Final values of the cost function (ΔE units) evaluated at the points in the training set at the end of optimization. (Numbers inside parentheses indicate percentage improvement over a uniform regular LUT of the same size.)

Type\Size	$6\times6\times6$	$9\times9\times9$	$12\times12\times12$
Uniform regular	6.21	3.79	2.79
Symmetric optimal	5.38 (13.36%)	3.39 (10.55%)	2.38 (14.69%)
Nonsymmetric optimal	4.87 (21.58%)	2.93 (22.69%)	2.19 (21.51%)

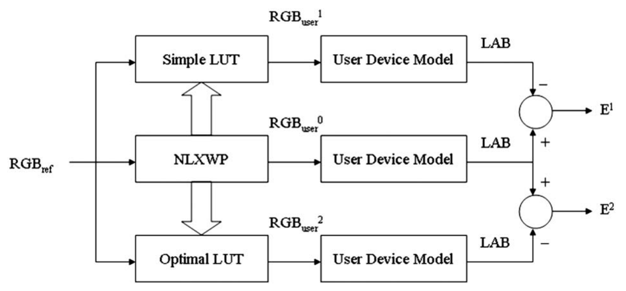

Figure 12. Block diagram for evaluating the optimal and nonoptimal LUTs against the NLXWP model.

fully design the training set S_T. By an intuitive crafting of S_T, one can instruct the system to acquire greater accuracy in the desired regions of the color space.

Let the optimal LUT be represented by $\Phi:\mathfrak{I}\rightarrow\mathfrak{I}$ such that $\Psi_{\text{reference}}(i)\approx\Psi_{\text{user}}[\Phi(i)]$ for any $i\in\mathfrak{I}$. We would like to compare the visual similarity achieved by a LUT-based approach with that achieved by a profile-based approach. Typical ICC profiles can range from 500 bytes to over 500 Kbytes[61] and a profile-based color management system would use two such profiles as shown in Fig. 2. Therefore, for a fair comparison we assume that our LUT can be as large as two ICC profiles. We perform our comparative experiments using profiles of two sizes—7 and 66 Kbytes. The process of generating these profiles and obtaining a color transformation using them is described in the next section.

EXPERIMENTAL RESULTS

We evaluate the effectiveness of the proposed color management system in two parts. First, the error statistics of the optimal LUT-based system are obtained by comparing the colors predicted by the LUT against those predicted by the NLXWP model which is taken as the "gold standard" reference. Some of these results were presented in earlier publications.[41,62] In the second part, we compare the LUT-based system with a profile-based system, and here we do not use the NLXWP as a reference. We note that our original problem was to obtain a visual output on a given display device (*user device*) that matches the visual output on a *reference device*. We use the simulated reference device model to obtain the reference (or desired) visual output for given RGB inputs. Similarly, we obtain the visual outputs for the color-corrected RGB inputs (as given by the LUT and the profile-based function) using the simulated user device model. The closeness of the two outputs to the reference would indicate the success of the corresponding color-management approach.

Table III. Testing error statistics (ΔE units) for LUTs evaluated relative to the NLXWP model. (Simple LUTs use separable inverse distance interpolations and a uniform sampling of the RGB space, and numbers inside the parentheses represent percentage improvement as obtained by an optimal LUT.)

Type\Statistic	Mean	Median	1-Std. Dev.
$6\times6\times6$ simple (E^1)	8.71	7.31	6.81
$6\times6\times6$ optimal (E^2)	4.84 (44.43%)	4.21	3.47
$9\times9\times9$ simple (E^1)	5.49	4.69	4.72
$9\times9\times9$ optimal (E^2)	2.92 (46.81%)	2.49	2.39
$12\times12\times12$ simple (E^1)	3.96	3.28	3.80
$12\times12\times12$ optimal (E^2)	2.19 (44.70%)	1.85	1.98

We provide the results of the training process described in the section Efficient Realization Using Look-Up Tables for LUTs of three sizes in Table II. These were obtained for a 1000 point training data set as stated earlier. Therefore, the values can be seen as the minimum *cost* at the end of the optimization process as shown in Fig. 11 (although there is no optimization in the case of uniform LUT, we can still view the value as the cost after a single iteration). Optimization was done for both symmetric and nonsymmetric LUTs and the numbers inside the respective parentheses indicate the percentage reduction in the average prediction error as compared to a LUT of the same size but with uniformly sampled Ω. Note that the optimization is carried out only with respect to sampling and all three types of LUTs use the same (separable squared inverse distance) interpolation scheme.

These results validate the applicability of our optimization technique to the problem of generating optimal LUTs for color management. It can be seen that by optimizing just one variable in the LUT-based system (sampling of the RGB space), we can reduce the average error by more than 20%. This is true even when the size of the LUT is as small as $6\times6\times6$. We further observe that addition of constraints on the optimization problem (resulting in symmetric LUTs) still reduces the average error by over 10% while significantly reducing the size of the problem as described in the section Efficient Realization Using Look-Up Tables. In the next experiment we test the optimal LUTs against simple LUTs which do not use any optimized parameters.

Figure 12 shows our method for evaluating the accuracy of the optimal LUT. Similar to Fig. 8, the block arrows represent the offline process of constructing and training the LUTs. The optimal LUTs (which use optimal sampling and separable squared inverse distance interpolation) were tested relative to the NLXWP function using another 1000 point data set, the results are presented in Table III. The statistics correspond to the errors denoted by E^2 in Fig. 12. The function was also approximated by nonoptimized LUTs (represented as "$n\times n\times n$ Simple" in the table and producing errors denoted by E^1 in the figure) which used separable inverse distance interpolations on a uniformly sampled Ω. While the (R,G,B) values used in the testing set were also randomly generated from the RGB space, we verified that the training and testing sets did not have any overlapping data points.

It can be observed from the statistics that the optimal LUTs generated by our system offers significant improvement over simple LUTs of the same size. Note that the mean values (column 2) are comparable to the cost function computed for the optimization, and hence, we compute the percentage reduction in this quantity as achieved by an optimal LUT over a simple LUT (provided inside parentheses). It is also evident that the effect of two independently selected methods (optimal sampling and interpolation) is cumulative and the percentage reduction in the average error exceeds the percentage reduction obtained in Table II where only the sampling method was being optimized. This completes our analysis of how well an optimal LUT can approximate a given function and we proceed to testing our optimal LUTs in a color management scenario.

At this point, we remind the reader that all our experiments were done on two virtual (or simulated) display devices. These devices were, essentially, highly accurate models of two real display devices (which are shown in Fig. 5). Generating the output with these models (for a given input RGB value) is equivalent to measuring the output obtained on the actual displays for the same input.

In order to compare the LUT-based system with a profile-based system, we require a function $g:\mathfrak{I}\rightarrow\mathfrak{I}$ which achieves perceptual similarity and is obtained from ICC-compatible device profiles. We used the open source software LPROF[63] to generate monitor profiles. The LPROF software is actively supported and is ICC version 2 compatible. It can generate a device profile based on selected parameters and a measurement sheet. The following steps are involved in obtaining the function g:

- We use the simulated ideal monitors to obtain the set of measurements required by the profiling software. This consists of computing the simulated CIE XYZ values obtained when different RGB inputs are used. Our set of RGB inputs contains grayscale, red, green, blue, and composite colors. The measurements for the two displays are recorded into corresponding "IT8" files.[64] Note that unlike a scanner calibration operation, we do not read the RGB values for an IT8 target. Instead we generate an IT8 target by passing known RGB inputs to the monitors and recording the output in CIE XYZ. For our experiments, we used 432 RGB values for creating the IT8 files. These consisted of 24 gray, 64 red, 64 green, 64 blue, and 216 mixed colors.
- LPROF uses the IT8 files as input and generates the corresponding "ICM" files. We specify the desired color LUT size and the chromatic adaptation model to be used as parameters while generating the profiles. The color reproduction intent for generating the profiles is specified as "perceptual."
- The profiles thus generated are processed using MATLAB with the MATLAB "iccread" function, and the color transformation g is obtained using the MATLAB "makecform" function. Note that the steps involved in creating the transformation g is similar to the steps involved in generating a device-link profile using the two device profiles as inputs. Hence, the accuracy of the color reproduction using such a device-link profile will not be significantly different from the accuracy when using the transformation g. Therefore, we did not perform separate experiments comparing our LUT-based approach with device-link profiles.

Table IV. Testing error statistics (ΔE units) for LUTs and ICC profiles.

Type\Statistic	Mean	Median	1-Std. Dev.	P_{95} [a]
ICC-1[b] (7 KB)	10.12	8.76	5.95	22.76
ICC-2[c] (66 KB)	10.24	8.91	5.90	22.82
$6\times6\times6$ optimal (0.65 KB)	9.42	8.88	4.36	15.85
$9\times9\times9$ optimal (2.14 KB)	8.15	8.02	3.77	12.92
$12\times12\times12$ optimal (5 KB)	7.86	7.74	3.81	12.61

[a]The 95th percentile of the distribution.
[b]Profiles were generated using Linear Bradford[65] chromatic transformation.
[c]Profiles were generated using CIECAM97s (Ref. 66) color adaptation models with simulated dark surroundings.

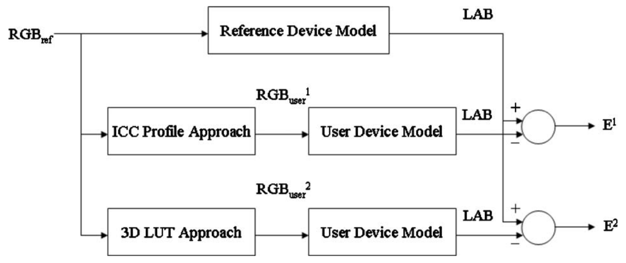

Figure 13. Block diagram for the comparison of a profile-based and a LUT-based system using the output of the simulated reference device as the reference. Here E^1 and E^2 are the errors obtained for the profile-based and the LUT-based approaches, respectively.

As illustrated in Figure 13, we compare the performance of the profile-based transform with that of the LUT-based transform by using the output of the simulated reference device as the benchmark. Note that this is different from the evaluation of the LUT when compared with NLXWP as shown in Fig. 12. There the output of the user device as predicted by NLXWP was taken as the benchmark, and hence, the error statistics may be different. Table IV summarizes the error statistics achieved by the profile-based transform and the LUT-based transform for various profile and LUT sizes. The numbers inside parentheses represent the memory required by the two approaches in kilobytes. In the case of profiles it is the size of the respective "ICM" files, while in the case of 3D LUT it is the memory required to store the LUT, and the sampling maps M_R, M_G and M_B as defined in Eqs. (18)–(20).

We observe that the LUT-based system provides more accurate color reproduction than a system that uses ICC profiles of much larger sizes. Furthermore, this observation is consistent in all the statistics—mean, median, one-standard deviation and the 95th percentile. This agrees with the intuition that a one-step transformation between two known devices can be achieved with greater success than a multistep transformation in which the devices are only characterized by their profiles. It is also seen that a larger optimal LUT provides better color reproduction accuracy than a smaller LUT. This is also not evident for profiles. Our aim in these experiments is to demonstrate the improved color reproduction offered by a LUT-based approach compared with profile-based approaches.

CONCLUSIONS

In this article we described a new method for color management of display devices. Unlike the conventional approach using ICC color profiles, we design a method based on 3D LUTs for transforming the digital data such that a perceptual match is achieved between the two devices. We believe that a 3D LUT will be faster than a profile-based method and is more suitable for hardware and embedded implementations. We have also shown in our experiments that a system using our 3D LUTs provides more accurate color reproduction than a profile-based system while requiring only a fraction of the memory needed to store an ICC profile. Our LUT-based color management system scales with the number of devices because each user device requires only one LUT for every reference device. The LUT-based system also provides the flexibility of selecting the LUT parameters for obtaining optimum color reproduction under specified resource constraints.

ACKNOWLEDGMENTS

This work was partially supported by a grant from the Indiana 21st Century Research and Technology Fund and by the endowment of the Charles William Harrison Distinguished Professorship at Purdue University.

REFERENCES

[1] B. Fraser, C. Murphy, and F. Bunting, *Real World Color Management* (Peachpit Press, Berkeley, 2004).

[2] G. Wyszecki and W. S. Stiles, *Color Science: Concepts and Methods, Quantitative Data, and Formulas* (Wiley-Interscience, New York, 2000).

[3] G. Gill, “What’s wrong with the ICC profile format anyway?”, http://www.argyllcms.com/icc_problems.html, accessed February 2009.

[4] P. Green, “New developments in ICC color management”, *Digital Futures 2008: International Standards around Printing and Imaging*, London, UK, October 2008.

[5] N. Koren, “Color management and color science: Introduction”, http://www.normankoren.com/color_management.html, accessed January 2009.

[6] D. Richardson, “Firefox 3: Color profile support”, http://www.dria.org/wordpress/archives/2008/04/29/633/, accessed January 2009.

[7] International Color Consortium, “File formats for color profiles”, ICC.1:2001–04, Reston, Virginia, 2001.

[8] Microsoft Corporation, “Windows Color System: The next generation color management system”, Microsoft White Paper: http://www.microsoft.com/whdc/device/display/color/WCS.mspx, September 2005.

[9] M. D. Fairchild, “A color scientist looks at video”, Proc. International Workshop on Video Processing and Quality Metrics for Consumer Electronics, (Scottsdale, Arizona, January 2007).

[10] Digital Cinema Initiative, LLC, “Digital cinema system specification”, Adopted and released document, version 1.2, Hollywood, California, 2008.

[11] G. Kennel, *Color and Mastering for Digital Cinema* (Focal Press, St. Louis, Missouri, 2006).

[12] R. W. G. Hunt, *The Reproduction of Color* (Wiley, New York, 2004).

[13] J. V. Kries, *Chromatic Adaptation* (Festschrift der Albrecht-Ludwig-Universitat, Friboug, Switzerland, 1902).

[14] International Color Consortium, “The role of ICC profiles in a color reproduction system”, ICC White Paper: http://www.color.org/ICC_white_paper_7_role_of_ICC_profiles.pdf, December 2004.

[15] A. G. Heidelberger Druckmaschinen, “Generation and application of devicelink profiles”, Prinect Color Solutions: http://www.heidelberg.com/www/html/en/binaries/files/prinect/device link profile pdf, accessed February 2010.

[16] Microsoft Corporation, “Using device profiles with WCS”, http://msdn.microsoft.com/en-us/library/dd372213%28VS.85%29.aspx, accessed February 2010.

[17] M. Maria, “Little CMS engine”, http://www.littlecms.com/TUTORIAL.TXT, accessed February 2010.

[18] M. R. Balonen-Rosen and J. E. Thornton, “User-interactive corrective tuning of color profiles”, US Patent 6,307,961 (2001).

[19] A. G. Colour Science, “Test images for monitor and printer calibration”, http://www.colour-science.com/, accessed October 2008.

[20] O. Arslan, Z. Pizlo, and J. P. Allebach, “CRT calibration techniques for better accuracy including low-luminance colors”, Proc. SPIE **5293**, 18–22 2004.

[21] B. Bastani, R. Ghaffari, and B. Funt, “Optimal linear RGB to *XYZ* mapping for color display calibration”, *Proc. IS&T/SID 12th Color Imaging Conference* (IS&T, Springfield, VA, 2004) pp. 223–227.

[22] G. Sharma, “LCDs versus CRTs: Color-calibration and gamut considerations”, Proc. IEEE **90**, 605–622 (2002).

[23] Adobe Systems, Inc., “Matching RGB color from monitor to printer”, Technical Note 5122, San Jose, California, 1992.

[24] Commission Internationale de’lEclairage, Technical Committee 8, *Chromatic Adaptation under Mixed Illumination Condition when Comparing Softcopy and Hardcopy Images*, Vienna, Austria, CIE 162:2010.

[25] R. S. Gentile, E. Walowit, and J. P. Allebach, “A comparison of techniques for color gamut mismatch compensation”, J. Imaging Technol. **16**, 176–181 (1990).

[26] J. Morovik, “To develop a universal gamut mapping algorithm”, Doctoral thesis, University of Derby, 1998.

[27] D. L. Post and S. C. Calhoun, “Further evaluation of methods for producing desired colors on CRT monitors”, Color Res. Appl. **25**, 90–104 (2000).

[28] T. H. Ha, S. Srivastava, E. J. Delp, and J. P. Allebach, “Model based methods for developing color transformation between two display devices”, *Proc. International Conference on Image Processing*, Cairo, Egypt, (IEEE, Piscataway, NJ, 2009) pp. 3793–3796.

[29] C. Poynton, *Digital Video and HDTV: Algorithms and Interfaces* (Morgan Kaufmann, San Francisco, 2003).

[30] R. W. G. Hunt and L. M. Winter, “Color adaptation in picture-viewing situations”, J. Photogr. Sci. **23**, 112–115 (1975).

[31] G. D. Finlayson and M. S. Drew, “White-point preserving color correction”, *Proc. IS&T/SID 5th Color Imaging Conference* (IS&T, Springfield, VA, 1997) pp. 258–261.

[32] E. D. Montag and M. D. Fairchild, “Evaluation of chroma clipping techniques for three destination gamuts”, *Proc. IS&T/SID 6th Color Imaging Conference* (IS&T, Springfield, VA, 1998) pp. 57–61.

[33] T. H. Ha, S. Srivastava, E. J. Delp, and J. P. Allebach, “Monitor characterization model using multiple nonsquare matrices for better accuracy”, *Proc. IS&T/SID 17th Color Imaging Conference* (IS&T, Springfield, VA, 2009) pp. 117–122.

[34] H. R. Kang, “3D look-up table with interpolation”, *Color Technology for Electronic Imaging Devices* (SPIE Press, Bellingham, Washington, 1997).

[35] P. C. Pugsley, “Color correcting image reproducing methods and apparatus”, US Patent 3,893,166 (1975).

[36] H. Kotera, H. Hayami, H. Tsuchiya, R. Kan, K. Yoshida, T. Shibata, and Y. Tsuda, “Color separating method and apparatus using statistical techniques”, US Patent 4,090,243 (1978).

[37] M. Abdulwahab, J. L. Burkhardt, and J. J. McCann, “Method of and apparatus for transforming color image data on the basis of an isotropic and uniform colorimetric space”, US Patent 4,839,721 (1989).

[38] J. J. McCann, “High-resolution color photographic reproductions”, Proc. SPIE **3025**, 53–59 1997.

[39] J. J. McCann, “Color spaces for color mapping”, J. Electron. Imaging **8**, 354–364 (1999).

[40] J. J. McCann, “Digital color transforms applied to fine art reproduction”,

Proc. International Conference on Imaging Science and Hardcopy, Guilin, China (RSESC, Beijing, China, 1995) pp. 377–379.
[41] S. Srivastava, T. H. Ha, E. J. Delp, and J. P. Allebach, "Generating optimal look-up tables to achieve complex color space transformations", *Proc. International Conference on Image Processing*, Cairo, Egypt, (IEEE Press, New York, 2009) pp. 1641–1644.
[42] R. Bala and V. Klassen, in *Digital Color Imaging Handbook*, edited by G. Sharma (CRC Press, Boca Raton, FL, 2002).
[43] D. Kidner, M. Dorey, and D. Smith, "What's the point? Interpolation and extrapolation on a regular grid DEM", Proceedings GeoComputation, 1999.
[44] D. Shepard, "A two-dimensional interpolation function for irregularly shaped data", *Proc. ACM National Conference* (ACM, Princeton, NJ, 1968), pp. 517–524.
[45] J. Z. Chan, J. P. Allebach, and C. A. Bouman, "Sequential linear interpolation of multidimensional functions", IEEE Trans. Image Process. **6**, 1231–1245 (1997).
[46] J. M. Kasson, S. I. Nin, W. Plouffe, and L. James, "Performing color space conversions with three dimensional linear interpolation", J. Electron. Imaging **4**, 226–250 (1995).
[47] C. D. Boor, *Guide to Splines* (Springer-Verlag, New York, 2001).
[48] H. Hou and H. Andrews, "Cubic splines for image interpolation and digital filtering", IEEE Trans. Acoust., Speech, Signal Process. **26**, 508–517 (1978).
[49] M. Unser, A. Aldroubi, and M. Eden, "B-spline signal processing: Part I: Theory", IEEE Trans. Signal Process. **41**, 821–833 (1993).
[50] M. Unser, A. Aldroubi, and M. Eden, "B-spline signal processing: Part II—Efficiency, design and applications", IEEE Trans. Signal Process. **41**, 834–848 (1993).
[51] I. E. Bell and W. Cowan, "Device characterization using spline smoothing and sequential linear interpolation", *Proc. IS&T/SID 2nd Color Imaging Conference* (IS&T, Springfield, VA, 1994) pp. 29–32.
[52] G. Sharma and M. Q. Shaw, "Thin-plate splines for printer data interpolation", *Proc. European Signal Processing Conference* (EURASIP, Kessariani, Greece, 2006).
[53] V. Monga and R. Bala, "Sort-select-damp: An efficient strategy for color look-up table lattice design", *Proc. IS&T/SID 16th Color Imaging Conference* (IS&T, Springfield, 2008) pp. 247–253.
[54] V. Monga and R. Bala, "Algorithms for color look-up-table (LUT) design via joint optimization of node locations and output values", *Proc. International Conference on Acoustics, Speech, and Signal Processing* (IEEE, Piscataway, NJ, 2010).
[55] R. M. Lewis and V. Torczon, "Pattern search methods for linearly constrained minimization", SIAM J. Optim. **10**, 917–941 (2000).
[56] R. M. Lewis, and V. Torczon, "Pattern search methods for bound constrained minimization", SIAM J. Optim. **9**, 1082–1099 (1999).
[57] V. Torczon, "On the convergence of pattern search algorithms", SIAM J. Optim. **7**, 1–25 (1997).
[58] The Mathworks Inc, "Finding minimum of functions using pattern search", http://www.mathworks.com/access/helpdesk/help/toolbox/gads/patternsearch.html, accessed February 2010.
[59] Sandia Corporation, "OPT++:an object-oriented nonlinear optimization library", https://software.sandia.gov/opt++/, accessed February 2010.
[60] E. Dolan and V. Torczon, "C++ class-based pattern search", http://www.cs.wm.edu/v~a/software/PatternSearch/, accessed February 2010.
[61] T. Newman, "Improved color for the world wide web: A case study in color management for distributed digital media", http://www.color.org/wpaper2.xalter, accessed March 2009.
[62] S. Srivastava, T. H. Ha, J. P. Allebach, and E. J. Delp, "Color management using device models and look-up tables", *Proc. Gjøvik Color Imaging Symposium*, (Gjøvik University College, Gjøvik, Norway, 2009) pp. 54–61.
[63] LPROF project team, "LPROF ICC profiler", http://lprof.sourceforge.net/, accessed April 2009.
[64] Committee for Graphics Arts Technologies Standards, "Graphic technology—color transmission target for input scanner calibration", ANSI Technical standard IT8.7/1–1993, Reston, Virginia, 2008.
[65] K. M. Lam, "Metamerism and color constancy", Doctoral thesis, University of Bradford, UK, 1985.
[66] M. R. Luo and R. W. G. hunt, "The structure of the CIE 1997 color appearance model (CIECAM97s)", Color Res. Appl. **23**, 138–146 (1998).