

Color Transfer between Images

Erik Reinhard, Michael Ashikhmin, Bruce Gooch,
and Peter Shirley
University of Utah

One of the most common tasks in image processing is to alter an image's color. Often this means removing a dominant and undesirable color cast, such as the yellow in photos taken under incandescent illumination. This article describes a method for a more general form of color correction that borrows one image's color characteristics from another. Figure 1 shows an example of this process, where we applied the colors of a sunset photograph to a daytime computer graphics rendering.

We use a simple statistical analysis to impose one image's color characteristics on another. We can achieve color correction by choosing an appropriate source image and apply its characteristic to another image.

We can imagine many methods for applying the colors of one image to another. Our goal is to do so with a simple algorithm, and our core strategy is to choose a suitable color space and then to apply simple operations there. When a typical three channel image is represented in any of the most well-known color spaces, there will be correlations between the different channels' values. For example, in RGB space, most pixels will have large values for the red and green channel if the blue channel is large. This implies that if we want to change the appearance of a pixel's color in a coherent way, we must modify all color channels in tandem. This complicates any color modification process. What we want is an orthogonal color space without correlations between the axes.

Recently, Ruderman et al. developed a color space, called $\alpha\beta$, which minimizes correlation between channels for many natural scenes.² This space is based on data-driven human perception research that assumes the human visual system is ideally suited for processing natural scenes. The authors discovered $\alpha\beta$ color space in the context of understanding the human visual system, and to our knowledge, $\alpha\beta$ space has never been applied otherwise or compared to other color spaces. There's little correlation between the axes in $\alpha\beta$ space,

which lets us apply different operations in different color channels with some confidence that undesirable cross-channel artifacts won't occur. Additionally, this color space is logarithmic, which means to a first approximation that uniform changes in channel intensity tend to be equally detectable.³

Decorrelated color space

Because our goal is to manipulate RGB images, which are often of unknown phosphor chromaticity, we first show a reasonable method of converting RGB signals to Ruderman et al.'s perception-based color space $\alpha\beta$. Because $\alpha\beta$ is a transform of LMS cone space, we first convert the image to LMS space in two steps. The first is a conversion from RGB to XYZ tristimulus values. This conversion depends on the phosphors of the monitor that the image was originally intended to be displayed on. Because that information is rarely available, we use a device-independent conversion that maps white in the chromaticity diagram (CIE xy) to white in RGB space and vice versa. Because we define white in the chromaticity diagram as $x = X/(X + Y + Z) = 0.333$, $y = Y/(X + Y + Z) = 0.333$, we need a transform that maps $X = Y = Z = 1$ to $R = G = B = 1$. To achieve this, we modified the XYZitu601-1 (D65) standard conversion matrix to have rows that add up to 1. The International Telecommunications Union standard matrix is

$$M_{itu} = \begin{bmatrix} 0.4306 & 0.3415 & 0.1784 \\ 0.2220 & 0.7067 & 0.0713 \\ 0.0202 & 0.1295 & 0.9394 \end{bmatrix} \quad (1)$$

By letting $M_{itu}x = (111)^T$ and solving for x , we obtain a vector x that we can use to multiply the columns of matrix M_{itu} , yielding the desired RGB to XYZ conversion:

$$\begin{bmatrix} X \\ Y \\ Z \end{bmatrix} = \begin{bmatrix} 0.5141 & 0.3239 & 0.1604 \\ 0.2651 & 0.6702 & 0.0641 \\ 0.0241 & 0.1228 & 0.8444 \end{bmatrix} \begin{bmatrix} R \\ G \\ B \end{bmatrix} \quad (2)$$

Compared with M_{itu} , this normalization procedure con-

stitutes a small adjustment to the matrix's values.

Once in device-independent XYZ space, we can convert the image to LMS space using the following conversion:⁴

$$\begin{bmatrix} L \\ M \\ S \end{bmatrix} = \begin{bmatrix} 0.3897 & 0.6890 & -0.0787 \\ -0.2298 & 1.1834 & 0.0464 \\ 0.0000 & 0.0000 & 1.0000 \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \end{bmatrix} \quad (3)$$

Combining these two matrices gives the following transformation between RGB and LMS cone space:

$$\begin{bmatrix} L \\ M \\ S \end{bmatrix} = \begin{bmatrix} 0.3811 & 0.5783 & 0.0402 \\ 0.1967 & 0.7244 & 0.0782 \\ 0.0241 & 0.1288 & 0.8444 \end{bmatrix} \begin{bmatrix} R \\ G \\ B \end{bmatrix} \quad (4)$$

The data in this color space shows a great deal of skew, which we can largely eliminate by converting the data to logarithmic space:²

$$\begin{aligned} \mathbf{L} &= \log L \\ \mathbf{M} &= \log M \\ \mathbf{S} &= \log S \end{aligned} \quad (5)$$

Using an ensemble of spectral images that represents a good cross-section of naturally occurring images, Ruderman et al. proceed to decorrelate these axes. Their motivation was to better understand the human visual system, which they assumed would attempt to process input signals similarly. We can compute maximal decorrelation between the three axes using principal components analysis (PCA), which effectively rotates them. The three resulting orthogonal principal axes have simple forms and are close to having integer coefficients. Moving to those nearby integer coefficients, Ruderman et al. suggest the following transform:

$$\begin{bmatrix} l \\ \alpha \\ \beta \end{bmatrix} = \begin{bmatrix} \frac{1}{\sqrt{3}} & 0 & 0 \\ 0 & \frac{1}{\sqrt{6}} & 0 \\ 0 & 0 & \frac{1}{\sqrt{2}} \end{bmatrix} \begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & -2 \\ 1 & -1 & 0 \end{bmatrix} \begin{bmatrix} \mathbf{L} \\ \mathbf{M} \\ \mathbf{S} \end{bmatrix} \quad (6)$$

If we think of the \mathbf{L} channel as red, the \mathbf{M} as green, and the \mathbf{S} as blue, we can see that this is a variant of many opponent-color models:⁴

$$\begin{aligned} \text{Achromatic} &\propto r + g + b \\ \text{Yellow-blue} &\propto r + g - b \\ \text{Red-green} &\propto r - g \end{aligned} \quad (7)$$

Thus the l axis represents an achromatic channel, while the α and β channels are chromatic yellow-blue and red-green opponent channels. The data in this space are symmetrical and compact, while we achieve decorrelation to higher than second order for the set of natural images tested.² Flanagan et al.⁵ mentioned this color space earlier because, in this color space, the achromatic axis is orthogonal to the equiluminant



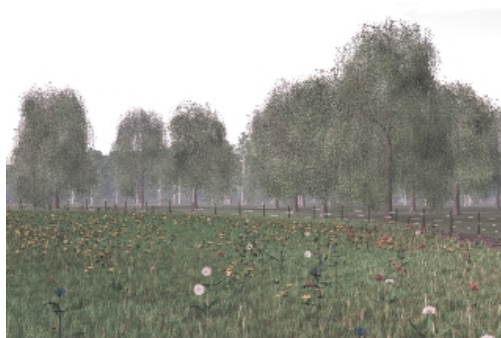
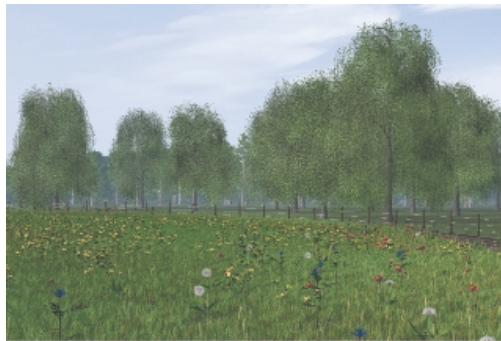
1 Applying a sunset to an ocean view. (a) Rendered image¹ (image courtesy of Simon Premoze), (b) photograph (image courtesy of the National Oceanic and Atmospheric Administration Photo Library), and (c) color-processed rendering.

plane. Our color-correction method operates in this $l\alpha\beta$ space because decorrelation lets us treat the three color channels separately, simplifying the method.

After color processing, which we explain in the next section, we must transfer the result back to RGB to display it. For convenience, here are the inverse operations. We convert from $l\alpha\beta$ to LMS using this matrix multiplication:

$$\begin{bmatrix} \mathbf{L} \\ \mathbf{M} \\ \mathbf{S} \end{bmatrix} = \begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & -1 \\ 1 & -2 & 0 \end{bmatrix} \begin{bmatrix} \frac{\sqrt{3}}{3} & 0 & 0 \\ 0 & \frac{\sqrt{6}}{6} & 0 \\ 0 & 0 & \frac{\sqrt{2}}{2} \end{bmatrix} \begin{bmatrix} l \\ \alpha \\ \beta \end{bmatrix} \quad (8)$$

2 Color correction in different color spaces. From left to right, the original source and target images, followed by the corrected images using RGB, $\alpha\beta$, and CIECAM97s color spaces. (Source image courtesy of Oliver Deussen.)



Then, after raising the pixel values to the power ten to go back to linear space, we can convert the data from LMS to RGB using

$$\begin{bmatrix} R \\ G \\ B \end{bmatrix} = \begin{bmatrix} 4.4679 & -3.5873 & 0.1193 \\ -1.2186 & 2.3809 & -0.1624 \\ 0.0497 & -0.2439 & 1.2045 \end{bmatrix} \begin{bmatrix} L \\ M \\ S \end{bmatrix} \quad (9)$$

Statistics and color correction

The goal of our work is to make a synthetic image take on another image's look and feel. More formally this means that we would like some aspects of the distribution of data points in $\alpha\beta$ space to transfer between images. For our purposes, the mean and standard deviations along each of the three axes suffice. Thus, we compute these measures for both the source and target images. Note that we compute the means and standard deviations for each axis separately in $\alpha\beta$ space.

First, we subtract the mean from the data points:

$$\begin{aligned} l^* &= l - \langle l \rangle \\ \alpha^* &= \alpha - \langle \alpha \rangle \\ \beta^* &= \beta - \langle \beta \rangle \end{aligned} \quad (10)$$

Then, we scale the data points comprising the synthetic image by factors determined by the respective standard deviations:

$$\begin{aligned} l' &= \frac{\sigma_t^l}{\sigma_s^l} l^* \\ \alpha' &= \frac{\sigma_t^\alpha}{\sigma_s^\alpha} \alpha^* \\ \beta' &= \frac{\sigma_t^\beta}{\sigma_s^\beta} \beta^* \end{aligned} \quad (11)$$

After this transformation, the resulting data points have standard deviations that conform to the photograph. Next, instead of adding the averages that we previously subtracted, we add the averages computed for the photograph. Finally, we convert the result back to RGB via log LMS, LMS, and XYZ color spaces using Equations 8 and 9.

Because we assume that we want to transfer one image's appearance to another, it's possible to select source and target images that don't work well together. The result's quality depends on the images' similarity in composition. For example, if the synthetic image contains much grass and the photograph has more sky in it, then we can expect the transfer of statistics to fail.

We can easily remedy this issue. First, in this example, we can select separate swatches of grass and sky and compute their statistics, leading to two pairs of clusters in $\alpha\beta$ space (one pair for the grass and sky swatches). Then, we convert the whole rendering to $\alpha\beta$ space. We scale and shift each pixel in the input image according to the statistics associated with each of the cluster pairs. Then, we compute the distance to the center of each of the source clusters and divide it by the cluster's standard deviation $\sigma_{c,s}$. This division is required to com-

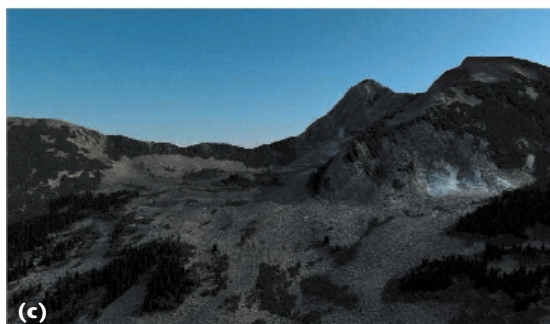
pensate for different cluster sizes. We blend the scaled and shifted pixels with weights inversely proportional to these normalized distances, yielding the final color. This approach naturally extends to images with more than two clusters. We can devise other metrics to weight relative contributions of each cluster, but weighting based on scaled inverse distances is simple and worked reasonably well in our experiments.

Another possible extension would be to compute higher moments such as skew and kurtosis, which are respective measures of the lopsidedness of a distribution and of the thickness of a distribution's tails. Imposing such higher moments on a second image would shape its distribution of pixel values along each axis to more closely resemble the corresponding distribution in the first image. While it appears that the mean and standard deviation alone suffice to produce practical results, the effect of including higher moments remains an interesting question.

Results

Figures 1 and 2 showcase the main reason for developing this technique. The synthetic image and the photograph have similar compositions, so we can transfer the photograph's appearance to the synthetic image. Figures 3 and 4 give other examples where we show that fairly dramatic transfers are possible and still produce believable results. Figure 4 also demonstrates the effectiveness of using small swatches to account for the dissimilarity in image composition.

Figure 5 (next page) shows that nudging some of the statistics in the right direction can sometimes make the result more visually appealing. Directly applying the method resulted in a corrected image with too much red in it. Reducing the standard deviation in the red–green channel by a factor of 10 produced the image in Figure 5c, which more closely resembles the old photograph's appearance.



3 Different times of day. (a) Rendered image⁶ (image courtesy of Simon Premoze), (b) photograph (courtesy of the NOAA Photo Library), and (c) corrected rendering.



4 Using swatches. We applied (a) the atmosphere of Vincent van Gogh's *Cafe Terrace on the Place du Forum, Arles, at Night* (Arles, September 1888, oil on canvas; image from the Vincent van Gogh Gallery, <http://www.vangoghgallery.com>) to (b) a photograph of Lednice Castle near Brno in the Czech Republic. (c) We matched the blues of the sky in both images, the yellows of the cafe and the castle, and the browns of the tables at the cafe and the people at the castle separately.

5 (a) Rendered forest (image courtesy of Oliver Deussen); (b) old photograph of a tuna processing plant at Sancti Petri, referred to as El Bosque, or the forest (image courtesy of the NOAA Photo Library); and (c) corrected rendering.



6 Color correction. Top to bottom: renderings using red, tungsten, and blue light sources. Left to right: original image, corrected using gray world assumption in $\alpha\beta$ space. The averages were browsed to more closely resemble Pattanaik's color correction results.⁷ (Images courtesy of Mark Fairchild.)



We show that we can use the $\alpha\beta$ color space for hue correction using the gray world assumption. The idea behind this is that if we multiply the three RGB channels by unknown constants c_r , c_g , and c_b , we won't change the $\alpha\beta$ channels' variances, but will change their means. Thus, manipulating the mean of the two chromatic channels achieves hue correction. White is specified in LMS space as (1, 1, 1), which converts to (0, 0, 0) in $\alpha\beta$ space. Hence, in $\alpha\beta$ space, shifting the average of the chromatic channels α and β to zero achieves hue correction. We leave the average for the achromatic channel unchanged, because this would affect the overall luminance level. The standard deviations should also remain unaltered.

Figure 6 shows results for a scene rendered using three different light sources. While this hue correction method overshoots for the image with the red light source (because the gray world assumption doesn't hold for this image), on the whole it does a credible job on removing the light source's influence. Changing the gray world assumption implies the chromatic α and β averages should be moved to a location other than (0, 0). By making that a 2D choice problem, it's easy to browse possibilities starting at (0, 0), and Figure 6's third column shows the results of this exercise. We hand calibrated the images in this column to resemble the results of Pattanaik et al.'s chromatic adaption algorithm,⁷ which for the purposes of demonstrating our approach to hue correction we took as ground truth.

Finally, we showcase a different use of the $\alpha\beta$ space in the sidebar "Automatically Constructing NPR Shaders."

Gamma correction

Note that we haven't mentioned gamma correction for any of the images. Because the mean and standard deviations are manipulated in $\alpha\beta$ space, followed by a transform back to LMS space, we can obtain the same results whether or not the LMS colors are first gamma corrected. This is because $\log x^\gamma = \gamma \log x$, so gamma cor-

Automatically Constructing NPR Shaders

To approximate an object's material properties, we can also apply gleaming color statistics from image swatches to automatically generating nonphotorealistic shading (NPR) models in the style of Gooch et al.¹ Figure A shows an example of a shading model that approximates artistically rendered human skin.

Gooch's method uses a base color for each separate region of the model and a global cool and warm color. Based on the surface normal, they implement shaders by interpolating between the cool and the warm color with the regions' base colors as intermediate control points. The original method requires users to select base, warm, and cool colors, which they perform in YIQ space. Unfortunately, it's easy to select a combination of colors that yields poor results such as objects that appear to be lit by a myriad of differently colored lights.

Rademacher² extended this method by shading different regions of an object using different sets of base, cool, and warm colors. This solution can produce more pleasing results at the cost of even more user intervention.

Using our color statistics method, it's possible to automate the process of choosing base colors. In Figure A, we selected a swatch containing an appropriate gradient of skin tones from Lena's back.³ Based on the swatch's color statistics, we implemented a shader that takes the average of

the swatch as base color and interpolates along the α axis in $\alpha\beta$ space between $+\sigma_L^\alpha$ and $-\sigma_L^\alpha$. We chose interpolation along the α axis because a yellow–blue gradient occurs often in nature due to the sun and sky. We limited the range to one standard deviation to avoid excessive blue and yellow shifts.

Because user intervention is now limited to choosing an appropriate swatch, creating proper NPR shading models is now straightforward and produces credible results, as Figure A shows.



A An example of an automatically generated NPR shader. An image of a 3D model of Michaelangelo's *David* shaded with an automatically generated skin shader generated from the full-size Lena image.³

References

1. A. Gooch et al., "A Non-Photorealistic Lighting Model for Automatic Technical Illustration," *Computer Graphics (Proc. Siggraph 98)*, ACM Press, New York, 1998, pp. 447-452.
2. P. Rademacher, "View-Dependent Geometry," *Proc. Siggraph 99*, ACM Press, New York, 1999, pp. 439-446.
3. L. Soderberg, centerfold, *Playboy*, vol. 19, no. 11, Nov. 1972.

rection is invariant to shifting and scaling in logarithmic $\alpha\beta$ space.

Although gamma correction isn't invariant to a transform from RGB to LMS, a partial effect occurs because the LMS axes aren't far from the RGB axes. In our experience, failing to linearize the RGB images results in approximately a 1 or 2 percent difference in result. Thus, in practice, we can often ignore gamma correction, which is beneficial because source images tend to be of unknown gamma value.

Color spaces

To show the significance of choosing the right color space, we compared three different color spaces. The three color spaces are RGB as used by most graphics algorithms, CIECAM97s, and $\alpha\beta$. We chose the CIECAM97s color space because it closely relates to the $\alpha\beta$ color space. Its transformation matrix to convert from LMS to CIECAM97s⁷ is

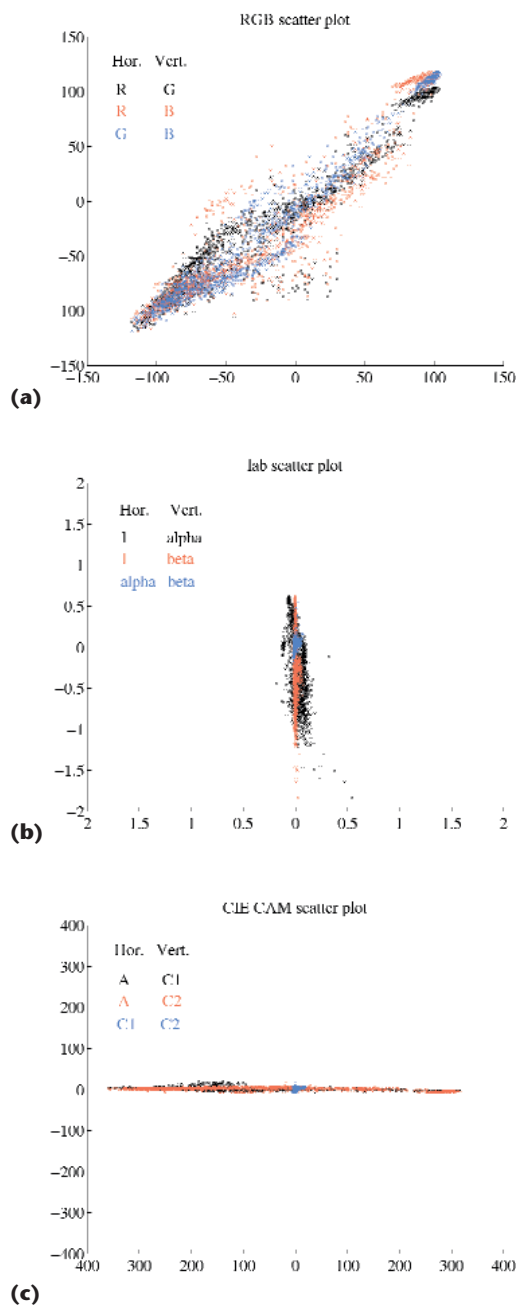
$$\begin{bmatrix} A \\ C_1 \\ C_2 \end{bmatrix} = \begin{bmatrix} 2.00 & 1.00 & 0.05 \\ 1.00 & -1.09 & 0.09 \\ 0.11 & 0.11 & -0.22 \end{bmatrix} \begin{bmatrix} L \\ M \\ S \end{bmatrix}$$

This equation shows that the two chromatic channels C_1 and C_2 resemble the chromatic channels in $\alpha\beta$ space,

bar a scaling of the axes. The achromatic channel is different (see Equation 6). Another difference is that CIECAM97s operates in linear space, and $\alpha\beta$ is defined in log space.

Using Figure 2's target image, we produced scatter plots of 2,000 randomly chosen data points in $\alpha\beta$, RGB, and CIECAM97s. Figure 7 (next page) depicts these scatter plots that show three pairs of axes plotted against each other. The data points are decorrelated if the data are axis aligned, which is the case for all three pairs of axes in both $\alpha\beta$ and CIECAM97s spaces. The RGB color space shows almost complete correlation between all pairs of axes because the data cluster around a line with a 45-degree slope. The amount of correlation and decorrelation in Figure 7 is characteristic for all the images that we tried. This provides some validation for Ruderman et al.'s color space because we're using different input data.

Although we obtained the results in this article in $\alpha\beta$ color space, we can also assess the choice of color space on our color correction algorithm. We expect that color spaces similar to $\alpha\beta$ space result in similar color corrections. The CIECAM97s space, which has similar chromatic channels, but with a different definition of the luminance channel, should especially result in similar images (perhaps with the exception of overall luminance). The most important difference is that



7 Scatter plots made in (a) RGB, (b) $\alpha\beta$, and (c) CIECAM97s color spaces. The image we used here is the target image of Figure 2. Note that the degree of correlation in these plots is defined by the angle of rotation of the mean axis of the point clouds, rotations of around 0 or 90 degrees indicate uncorrelated data and in between values indicate various degrees of correlation. The data in $\alpha\beta$ space is more compressed than in the other color spaces because it's defined in log space.

CIECAM97s isn't logarithmic. Figure 2 shows the results. Color correction in $\alpha\beta$ space produces a plausible result for the given source and target images. Using CIECAM97s space also produces reasonable results, but the image is more strongly desaturated than in $\alpha\beta$ space. It doesn't preserve the flowers' colors in the field

very well. Using RGB color space doesn't improve the image at all. This result reinforces the notion that the proper choice of color space is important.

Conclusions

This article demonstrates that a color space with decorrelated axes is a useful tool for manipulating color images. Imposing mean and standard deviation onto the data points is a simple operation, which produces believable output images given suitable input images. Applications of this work range from subtle postprocessing on images to improve their appearance to more dramatic alterations, such as converting a daylight image into a night scene. We can use the range of colors measured in photographs to restrict the color selection to ones that are likely to occur in nature. This method's simplicity lets us implement it as plug-ins for various commercial graphics packages. Finally, we foresee that researchers can successfully use $\alpha\beta$ color space for other tasks such as color quantization. ■

Acknowledgments

We thank everyone who contributed images to this article. We also thank the National Oceanic and Atmospheric Administration for putting a large photograph collection in the public domain. The National Science Foundation's grants 97-96136, 97-31859, 98-18344, 99-77218, 99-78099 and the US Department of Energy Advanced Visualization Technology Center (AVTC)/Views supported this work.

References

1. S. Premoze and M. Ashikhmin, "Rendering Natural Waters," *Proc. Pacific Graphics*, IEEE CS Press, Los Alamitos, Calif., 2000, pp. 23-30.
2. D.L. Ruderman, T.W. Cronin, and C.C. Chiao, "Statistics of Cone Responses to Natural Images: Implications for Visual Coding," *J. Optical Soc. of America*, vol. 15, no. 8, 1998, pp. 2036-2045.
3. D.R.J. Lamming, *Sensory Analysis*, Academic Press, London, 1986.
4. G. Wyszecki and W.S. Stiles, *Color Science: Concepts and Methods, Quantitative Data and Formulae*, 2nd ed., John Wiley & Sons, New York, 1982.
5. P. Flanagan, P. Cavanagh, and O.E. Favreau, "Independent Orientation-Selective Mechanism for The Cardinal Directions of Colour Space," *Vision Research*, vol. 30, no. 5, 1990, pp. 769-778.
6. S. Premoze, W. Thompson, and P. Shirley, "Geospecific Rendering of Alpine Terrain," *Proc. 10th Eurographics Workshop on Rendering*, Springer-Verlag, Wien, Austria, 1999, pp. 107-118.
7. S.N. Pattanaik et al., "A Multiscale Model of Adaptation and Spatial Vision for Realistic Image Display," *Computer Graphics (Proc. Siggraph 98)*, ACM Press, New York, 1998, pp. 287-298.



the University of Bristol.

Erik Reinhard is a postdoctoral fellow at the University of Utah in the visual-perception and parallel-graphics fields. He has a BS and a TWAIO diploma in computer science from Delft University of Technology and a PhD in computer science from



and Technology, an MS in chemistry from the University of California at Berkeley, and a PhD in computer science from the University of Utah.

Michael Ashikhmin is an assistant professor at State University of New York (SUNY), Stony Brook. He is interested in developing simple and practical algorithms for computer graphics. He has an MS in physics from the Moscow Institute of Physics



Bruce Gooch is a graduate student at the University of Utah. His research interest is in nonphotorealistic rendering. He has a BS in mathematics and an MS in computer science from the University of Utah. He's an NSF graduate fellowship awardee.



Illinois at Urbana-Champaign.

Peter Shirley is an associate professor at the University of Utah. His research interests are in realistic rendering and visualization. He has a BA in physics from Reed College, Portland, Oregon, and a PhD in computer science from the University of

Readers may contact Reinhard at the Dept. of Computer Science, Univ. of Utah, 50 S. Central Campus Dr. RM 3190, Salt Lake City, UT 84112-9205, email reinhard@cs.utah.edu.

For further information on this or any other computing topic, please visit our Digital Library at <http://computer.org/publications/dlib>.