# Overcoming Gamut and Dynamic Range Limitations in Digital Images

*Gregory Ward Larson*
*Silicon Graphics, Inc.*
*Mountain View, California*

## Abstract

The human eye can accommodate luminance in a single view over a range of about 10,000:1 and is capable of distinguishing about 10,000 colors at a given brightness. By comparison, typical CRT displays have a luminance range less than 100:1 and cover about half of the visible color gamut. Despite this difference, most digital image formats are geared to the capabilities of conventional displays, rather than the characteristics of human vision. In this paper, we propose two compact encodings suitable for the transfer, manipulation, and storage of full range color images. The first format is a replacement for conventional RGB images, and encodes color pixels as log luminance values and CIE (u',v') chromaticity coordinates. We have implemented and distributed this encoding as part of the standard TIFF I/O library on the net. The second format is proposed as an adjunct to conventional RGB data, and encodes out-of-gamut (and out-of-range) pixels in a supplemental image, suitable as a layer extension to the Flashpix standard. This data can then be recombined with the original RGB layer to obtain a high dynamic range image covering the full gamut of perceivable colors. Finally, we demonstrate the power and utility of full gamut imagery with example images and applications.

## Introduction

What is the ultimate use of a digital image? How will it be presented? Will it be modified or adjusted? What kind of monitor will it be displayed on? What type of printer will it be sent to? How accurate do the colors need to be? More often than not, we don't know the answers to these questions a priori. More important, we don't know how these questions will be answered 10 or 100 years from now, when everything we know about digital imaging will have changed, but someone may still want to use our image. We should therefore endeavor to record image data that will be valuable under a broad range of foreseeable and postulated circumstances. Although this seems problematic, there is a simple solution. We may not be able to predict the technology, but we can predict that people will still be the primary consumers.

Most commonly used image standards based on current display technology, i.e., CRT monitors, rather than something less apt to change, i.e., human vision. All RGB standards are limited to a fraction of the visible gamut, since this gamut cannot be contained between any three *real* colors. Even Kodak's PhotoYCC encoding is ultimately geared for CRT display, and doesn't encompass the full gamut of colors or cover more than two orders of magnitude in brightness. The human eye is capable of perceiving at least four orders of magnitude in a daylit scene, and adapting more gradually over seven *additional* orders of magnitude, which means that most digital images encode only a small fraction of what a human observer can see.

In this sense, negative photography is superior to digital imaging in its ability to capture the dynamic range of a scene. A typical, consumer-grade color negative film has about 5-8 f-stops of *exposure latitude*, meaning that it can capture regions of a scene that are $2^5$ to $2^8$ times brighter than the camera's exposure setting (or dimmer if the image is overexposed), and still have enough range left over to reproduce each region[*]. Of course, most prints do not make use of the full range, unless a photographer picks up a wand or a cutout in the darkroom, but its presence permits re-exposure during the printing process to optimize the appearance of salient features, such as a person's face.

[*] To compute the latitude of a film or recording medium, take the log to the base 2 of the total usable dynamic range, from darkest unique value to brightest, and subtract 5 f-stops, which is the approximate range required for a usable image. There are about 3.3 f-stops per order of magnitude.

The question to ask is this: in 10 years or 100 years, what medium will be preferred for old photographs, a digital image, or a negative? Unless we change the way digital images are encoded, the answer in most cases will be a negative. Even considering aging and degradation (processes that can be partially compensated), a negative has both superior resolution and greater dynamic range than an RGB or YCC image. This needn't be the case.

In this paper, we present a compact pixel encoding using a log representation of luminance and a CIE (u',v') representation of color. We call this a *LogLuv* encoding. A log luminance representation means that at any exposure level, there will be equal brightness steps between values. This corresponds well with human visual response, whose contrast threshold is constant over a wide range of adaptation luminances (Weber's law). For color, the use of an approximately uniform perceptual space enables us to record the full gamut of visible colors using step sizes that are imperceptible to the eye. The combination of these two techniques permits us to make nearly optimal use of the bits available to record a given pixel, so that it may be reproduced over a broad range of viewing conditions. Also, since we are recording the full visible gamut and dynamic range, the output or display device can be *anything* and we won't be able to detect any errors or artifacts from our representation, simply because they will be reproduced below the visible threshold.

In this paper, we describe our LogLuv pixel encoding method, followed by a description of our extension to Sam Leffler's free TIFF library. We then put forth a proposal for extending the Flashpix format, and follow this with an example image to demonstrate the value of this encoding, ending with a brief conclusion.

## Encoding Method

We have implemented two LogLuv pixel encodings, a 24-bit encoding and a 32-bit encoding. The 24-bit encoding breaks down into a 10-bit log luminance portion and a 14-bit, indexed uv coordinate mapping. Color indexing minimizes waste, allowing us to cover the irregular shape of the visible gamut in imperceptible steps. The 32-bit encoding uses 16 bits for luminance and 8 bits each for u' and v'. Compared to the 24-bit encoding, the 32-bit version provides greater dynamic range and precision at the cost of an extra byte per pixel. The exact interpretations of these two encodings are described below.

### 24-bit Encoding

In 24 bits, we can pack much more visible information than is commonly stored in three gamma-compressed 8-bit color primary values. By separating luminance and using a log encoding, we can use 10 bits to record nearly 5 orders of magnitude in 1.1% relative steps that will be imperceivable under most conditions. The remaining 14 bits will be used to store a color index corresponding to the smallest distinguishable patch size on a uv color chart. The bit allocation is shown graphically in Fig. 1.

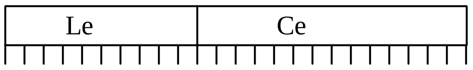

*Figure 1. 24-bit encoding. $L_e$ is the encoded log luminance, and $C_e$ is the encoded uv color index.*

To compute the integer encoding $L_e$ from real luminance, $L$, the we use the formula given in Eq. 1a. To compute real luminance from $L_e$, we use the inverse formula given in Eq. 1b.

$$L_e = \left\lfloor 64(\log_2 L + 12) \right\rfloor \qquad (1a)$$

$$L = \exp_2\left[(L_e + 0.5)/64 - 12\right] \qquad (1b)$$

In addition, an $L_e$ value of 0 is taken to equal 0.0 exactly. An $L_e$ value of 1 corresponds to a real luminance value of 0.000248 on an arbitrary scale, and the maximum $L_e$ value of 1023 corresponds to a real value of 15.9 for a dynamic range of 65,000:1, or 4.8 orders of magnitude. It is difficult to compare this range to an 8-bit gamma-compressed encoding, because 1.1% accuracy is possible only near the very top of the 8-bit range. Allowing the luminance error to go as high as 5%, the dynamic range of an 8-bit encoding with a nominal gamma of 2.2 is 47:1, or 1.7 orders of magnitude. This leaves less than one f-stop of exposure latitude, compared to 11 f-stops for our 10-bit log encoding.

To capture full-gamut chrominance using only 14 bits, we cannot afford to waste codes on imaginary colors. We therefore divide our "perceptually uniform" (u',v') color space [8] into equal area regions using a scanline traversal over the visible gamut. This encoding concept is shown graphically in Fig. 2. The actual encoding has many more scanlines of course (163 to be exact), but the figure shows roughly how they are laid out. The minimum code value (0) is at the lower left, and codes are assigned left to right along each scanline until the maximum value (just less than $2^{14}$) is assigned to the rightmost value on the top scanline.

$$u' = \frac{4x}{-2x + 12y + 3} \qquad (2a)$$

$$v' = \frac{9y}{-2x + 12y + 3} \qquad (2b)$$

To encode a given color, we start with the standard conversion from CIE (x,y) chromaticity to (u',v') shown in Eq. 2. We then look up the appropriate scanline for our *v'* value based on a uniform scanline height, and compute the position within the scanline using our uniform cell width. The index $C_e$ is equal to the total of the scanlines below us plus the cells to the left in this scanline. Cell width and height are both set to 0.0035 in our implementation, which corresponds to slightly less than the minimum perceptible step in this color space and uses up nearly all of the codes available in 14 bits.

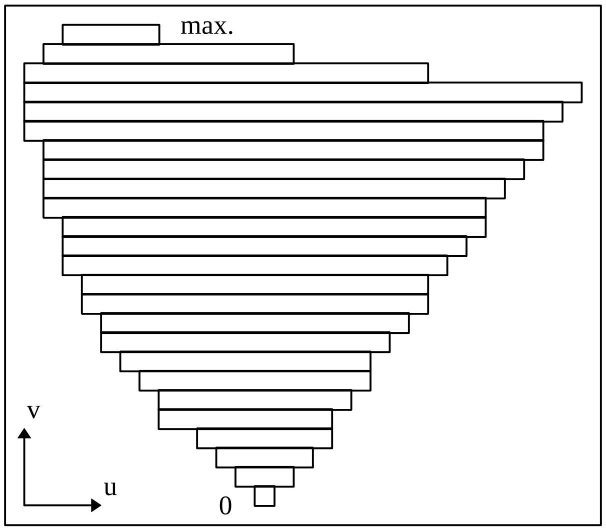

*Figure 2. Scanline traversal of (u,v) coordinate space for 14-bit chromaticity encoding.*

To get back the (x,y) chromaticity corresponding to a specific color index, we may either use a 16 Kentry look-up table, or apply a binary search to find the scanline containing corresponding to our $C_e$ index. Once we have our original (u',v') coordinates back, we can apply the inverse conversion given in Eq. 3 to get the CIE chromaticity coordinates. (Note that this final computation may also be avoided using the same look-up table.)

$$x = \frac{9u'}{6u' - 16v' + 12} \qquad (3a)$$

$$y = \frac{4v'}{6u' - 16v' + 12} \qquad (3b)$$

**32-bit Encoding**

The 32-bit encoding is actually simpler, since we have 16 bits for (u',v'), which is more than enough that we can dispense with the complex color indexing scheme. The encoding of luminance is similar, with the addition of a sign bit so that negative luminances may also be encoded. In the remaining 15 bits, we can record over 38 orders of magnitude in 0.27% relative steps, covering the full range of perceivable world luminances in imperceptible steps. The bit breakdown is shown in Fig. 3.

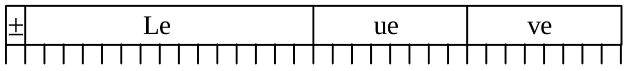

*Figure 3. Bit allocation for 32-bit pixel encoding. MSB is a sign bit, and the next 15 bits are used for a log luminance encoding. The uv coordinates are separate 8-bit quantities.*

The conversion to and from our log luminance encoding is given in Eq. 4. The maximum luminance using this encoding is $1.84\times10^{19}$, and the smallest magnitude is $5.44\times10^{-20}$. As in the 10-bit encoding, an $L_e$ value of 0 is taken to be exactly 0.0. The sign bit is extracted before encoding and reapplied after the conversion back to real luminance.

$$L_e = \lfloor 256(\log_2 L + 64) \rfloor \qquad (4a)$$

$$L = \exp_2\left[\left(L_e + 0.5\right)/256 - 64\right] \qquad (4b)$$

As we mentioned, the encoding of chrominance is simplified because we have enough bits to record $u_e$ and $v_e$ separately. Since the gamut of $u$ and $v$ values is between 0 and 0.62, we chose a scale factor of 410 to go between our [0,255] integer range and real coordinates, as given in Eq. 5.

$$u_e = \lfloor 410u' \rfloor \qquad (5a)$$

$$v_e = \lfloor 410v' \rfloor \qquad (5b)$$

$$u' = \left(u_e + 0.5\right)/410 \qquad (5c)$$

$$v' = \left(v_e + 0.5\right)/410 \qquad (5d)$$

This encoding captures the full color gamut in 8 bits each for $u_e$ and $v_e$. There will be some unused codes outside the visible gamut, but the tolerance this gives us of 0.0017 units in uv space is already well below the visible threshold. Conversions to and from CIE (x,y) chromaticities are the same as given earlier in Eqs. 2 and 3.

## TIFF Input/Output Library

The LogLuv encodings described have been embedded as a new `SGILOG` compression type in Sam Leffler's popular TIFF I/O library. This library is freely distributed by anonymous ftp on ftp.sgi.com in the "/graphics/tiff/" directory.

When writing a high dynamic range (HDR) TIFF image, the LogLuv *codec* (compression/decompresson module) takes floating point CIE XYZ scanlines and writes out 24-bit or 32-bit compressed LogLuv-encoded values. When reading an HDR TIFF, the reverse conversion is performed to get back floating point XYZ values. (We also provide a simple conversion to 24-bit gamma-compressed RGB for the convenience of readers that do not know how to handle HDR pixels.)

An additional tag is provided for absolute luminance calibration, named `TIFFTAG_STONITS`. This is a single floating point value that may be used to convert Y values returned by the reader to absolute luminance in candelas per square meter. This tag may also be set by the application that writes out a HDR TIFF to permit calibrated scaling of values to a reasonable brightness range, where values of 1.0 will be displayed at the maximum output of the destination device. This scale factor may also necessary for calibration of the 24-bit format due to its more limited dynamic range.

### Run-length Compression

Although at first it may appear that the 24-bit code is a more compact representation, the 32-bit encoding offers some advantages when it comes to applying nondestructive techniques to reduce storage requirements. By separating the bytes into four streams on each scanline, the 32-bit encoding can be efficiently compressed using an adaptive run-length encoding [3]. Since the top byte containing the sign bit and upper 7 log luminance bits changes very slowly, this byte-stream submits very well to run-length encoding. Likewise, the encoded $u_e$ and $v_e$ byte-streams compress well over areas of constant color. In contrast, the 24-bit encoding does not have a nice byte-stream breakup, so we do not attempt to run-length encode it, and the resulting files are quite often larger than the same data stored in the 32-bit format.

### Grayscale Images

For maximum flexibility, a pure luminance mode is also provided by the codec, which stores and retrieves run-length encoded 16-bit log luminance values using the same scheme as applied in the 32-bit LogLuv encoding. There is no real space savings over a straight 32-bit encoding, since the $u_e$ and $v_e$ byte-streams compress to practically nothing for grayscale data, but this option provides an explicit way to specify floating point luminance images for TIFF readers that care.

### Raw I/O

It is also possible to decode the raw 24-bit and 32-bit LogLuv data retrieved from an HDR TIFF directly, and this has some advantages for implementing fast tone mapping and display algorithms. In the case of the 24-bit format, one can simply multiply the output of a 1 Kentry $L_e$ table and a 16 Kentry $C_e$ table to get a tone-mapped and gamma-compressed RGB result. The 32-bit encoding requires a little more work, since its precomputed tables are 32 and 64 Kentries, but the same logic applies.

We have implemented this type of integer-math tone-mapping algorithm in an HDR image viewer, and it takes about a second to load and display a 512 by 512 picture on a 180 MHz processor.

### Example TIFF Code and Images

Use of this encoding is demonstrated and sample images are provided on the following web site:

*http://www.sgi.com/Technology/pixformat/*

A converter has been written to and from the *Radiance* floating point picture format [6][7], and serves as an example of LogLuv codec usage. The web site itself also offers programming tips and example code segments.

Example TIFF images using the 32-bit LogLuv and 16-bit LogL encoding are provided on the web site. These images are either scanned from photographic negatives or rendered using *Radiance* and converted to the new TIFF format. Some images are rendered as 360° QuickTime VR panoramas suitable for experiments in HDR virtual reality.

## Proposed Extension to Flashpix

The *Flashpix* format was originally developed by Kodak in collaboration with Hewlett-Packard, Live Picture and Microsoft. Its definition and maintenance has since been taken over by the Digital Imaging Group, a consortium of these and other companies. Flashpix is basically a multiresolution JPEG encoding, optimized for quick loading and editing at arbitrary pixel densities. It supports standard RGB as well as YCC color spaces with 8 bits/primary maximum resolution. For further information, see the DIG web site:

*http://www.digitalimaging.org*

Because Flashpix starts with 8-bit gamma-compressed color primaries, the dynamic range is limited to the same 1.7 orders of magnitude provided by other 24-bit RGB encodings. Furthermore, since JPEG encoding is applied, there will be additional losses and artifacts depending on the source image and the compression quality setting.

We cannot directly replace the JPEG-encoded Flashpix image with our own, alternate format, since this would violate standard compatibility as put forth by Kodak and enforced by the DIG. We must therefore provide any enhancement to the format as an optional extension, which results in a certain amount of redundancy in our case since the same pixels may be represented by two encodings. This is unavoidable.

For our extension, we need a second layer of “deeper” image data be provided for Flashpix users and applications that demand it. There are two ways we might go about this. The simplest method is to completely duplicate the source image in a 24 or 32-bit/pixel LogLuv encoding. On average, this will take roughly four to sixteen times as much space as the original JPEG encoding. A more sophisticated method is to replace only those pixels that are out of gamut or otherwise inadequate in the original encoding. We discuss this method below.

**High Dynamic Range Extension Layer**

Our proposed extension consists of a layer added to the standard Flashpix format. This layer contains two logical elements, a *presence map* of which pixels are included in the layer, and the list of corresponding 24-bit LogLuv pixels. The presence map may be represented by an entropy-encoded bitmap, which will typically take up 5% to 15% as much space as the JPEG layer. The extended pixels themselves will take between one half and four times as much space as the original JPEG layer, depending on the proportion of out-of-gamut pixels in the original image.

For an image that is entirely within gamut in the JPEG encoding, the presence map will compress to almost nothing, and there will be no LogLuv pixels, so the total overhead will be less than 1% of the original image. If the image is mostly out of the JPEG gamut, then the presence map might take half a bit per pixel, and the additional data will be the same size as a 24-bit RGB image. A typical high dynamic range image with 15% out-of-gamut pixels will take roughly the same space for the extension layer as the multiresolution JPEG layer, so the total image size will be about twice what it was originally. If the information is being accessed over the internet, the HDR layer may be loaded as an option, so it does not cost extra unless and until it is needed.

## Example Results

Fig. 4a shows a scanned photograph as it might appear on a PhotoCD using a YCC encoding. Since YCC can capture up to "200% reflectance," we can apply a tone mapping operator to bring this extra dynamic range into our print, as shown in Fig. 5a. However, since many parts of the image were brighter than this 200% value, we still lose much of the sky and circumsolar region, and even the lighter asphalt in the foreground. In Fig. 4b, we see where 35% of the original pixels are outside the gamut of a YCC encoding.

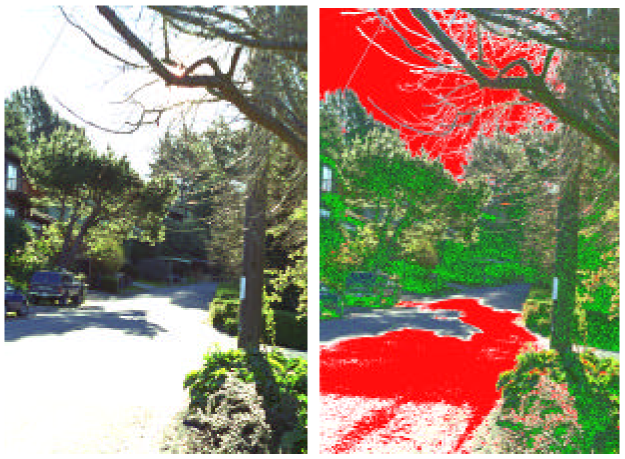

*Figure 4. The left image (a) shows a PhotoYCC encoding of a color photograph tone-mapped with a linear operator. The right image (b) shows the out-of-gamut regions. Red areas are too bright or too dim, and green areas have inaccurate color.*

Fig. 5b shows the same color negative scanned into our 32-bit/pixel high dynamic range TIFF format and tone mapped using a histogram compression technique [4]. Fig. 6c shows the same HDR TIFF remapped using the perceptual model of Pattanaik et al [5]. Figs. 6a and 6b show details of light and dark areas of the HDR image whose exposure has been adjusted to show the detail captured in the original negative. Without an HDR encoding, this information is either lost or unusable.

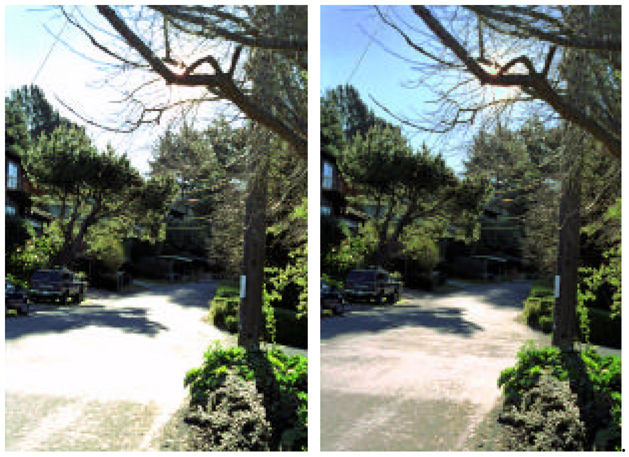

*Figure 5. The left image(a) shows the YCC encoding after remapping with a high dynamic range tone operator [4]. Unfortunately, since YCC has so little dynamic range, most of the bright areas are lost. The right image (b) shows the same operator applied to a 32-bit HDR TIFF encoding, showing the full dynamic range of the negative.*

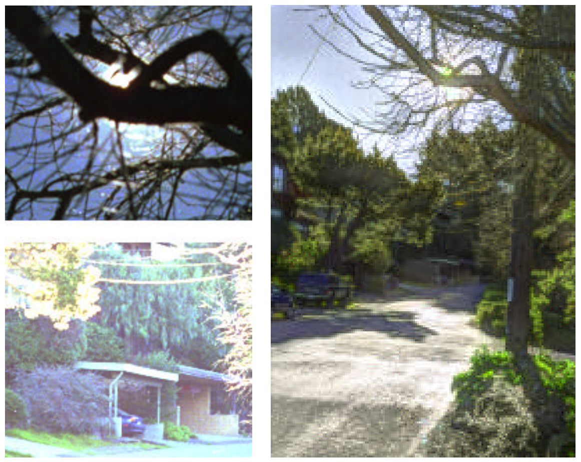

*Figure 6. The upper-left image (a) shows the circumsolar region reduced by 4 f-stops to show the image detail recorded on the negative. The lower-left image (b) shows house details boosted by 3 f-stops. The right image (c) shows our HDR TIFF mapped with the Pattanaik-Ferwerda tone operator [5].*

**Discussion**

It is clear from looking at these images that current methods for tone-mapping HDR imagery, although better than a simple S-curve, are less than perfect. It would therefore be a mistake to store an image that has been irreversibly tone mapped in this fashion, as some scanner software attempts to do. Storing an HDR image allows us to take full advantage of future improvements in tone mapping and display algorithms, at a nominal cost.

Besides professional photography, there are a number of application areas where HDR images are key. One is lighting simulation, where designers need to see an interior or exterior space as it would really appear, plus they need to evaluate things in terms absolute luminance and illuminance levels. Since an HDR image can store the real luminance in its full-gamut coverage, this information is readily accessible to the designer. Another application is image-based rendering, where a user is allowed to move about in a scene by warping captured or rendered images [1]. If these images have limited dynamic range, it is next to impossible to adapt the exposure based on the current view, and quality is compromised. Using HDR pixels, a natural view can be provided for any portion of the scene, no matter how bright or how dim. A fourth application area is digital archiving, where we are making a high-quality facsimile of a work of art for posterity. In this case, the pixels we record are precious, so we want to make sure they contain as much information as possible. At the same time, we have concerns about storage space and transmission costs, so keeping this data as compact as possible is important. Since our HDR format requires little more space than a standard 24-bit encoding to capture the full visible gamut, it is a clear winner for archiving applications.

Our essential argument is that we can make better use of the bits in each pixel by adopting a perceptual encoding of color and brightness. Although we don't know how a given image might be used or displayed in the future, we do know something about what a human can observe in a given scene. By faithfully recording this information, we ensure that our image will take full advantage of any future improvements in imaging technology, and our basic format will continue to find new uses.

## Conclusion

We have presented a new method for encoding high dynamic range digital images using log luminance and uv chromaticity to capture the entire visible range of color and brightness. The proposed format requires little additional storage per pixel, while providing significant benefits to suppliers, caretakers and consumers of digital imagery.

Through the use of re-exposure and dynamic range compression, we have been able to show some of the benefits of HDR imagery. However, it is more difficult to illustrate the benefits of a larger color gamut without carefully comparing hard copy output of various multi-ink printers. Also, since we currently lack the ability to capture highly saturated scenes, our examples would have to be contrived from individual spectral measurements and hypothetical scenes. We therefore leave this as a future exercise.

Future work on the format itself should focus on the application of lossy compression methods (such as JPEG and fractal image encoding) for HDR images. Without such methods, the storage cost for a given resolution may hinder broad acceptance of this representation. Another extension we should look at is multispectral data, which is needed for remote imaging and some types of lighting simulation.

## References

1. Paul Debevec, "Rendering Synthetic Objects into Real Scenes: Bridging Traditional and Image-Based Graphics with Global Illumination and High Dynamic Range Photography," *Computer Graphics (Proceedings of ACM Siggraph 98)*.
2. Paul Debevec, Jitendra Malik, "Recovering High Dynamic Range Radiance Maps from Photographs," *Computer Graphics (Proceedings of ACM Siggraph 97)*.
3. Andrew Glassner, "Adaptive Run-Length Encoding," in *Graphics Gems II*, edited by James Arvo, Academic Press, (1991).
4. Greg Larson, Holly Rushmeier, Christine Piatko, "A Visibility Matching Tone Reproduction Operator for High Dynamic Range Scenes," *IEEE Transactions on Visualization and Computer Graphics*, **3**, 4, (1997).
5. Sumant Pattanaik, James Ferwerda, Mark Fairchild, Don Greenberg, "A Multiscale Model of Adaptation and Spatial Vision for Realistic Image Display," *Computer Graphics (Proceedings of Siggraph 98)*.
6. Greg Ward, "The RADIANCE Lighting Simulation and Rendering System," *Computer Graphics (Proceedings of Siggraph 94)*.
7. Greg Ward, "Real Pixels," in *Graphics Gems II*, edited by James Arvo, Academic Press, (1991).
8. Gunter Wyszecki, W.S. Stiles, *Color Science: Concepts and Methods, Quantitative Data and Formulae*, Second Edition, Wiley, (1982).

## Biography

Gregory Ward Larson is a member of the technical staff in the engineering division of SGI. He graduated with an AB in Physics in 1983 from the UC Berkeley, and earned his Master's in CS from San Francisco State in 1985. Greg has done work in physically-based rendering, surface reflectance measurements, and electronic data standards. He is the developer of the widely-used *Radiance* synthetic imaging system and the MGF exchange standard for scene data.

Greg may be reached by e-mail at *gregl@sgi.com*.