JPEG-HDR: A Backwards-Compatible, High Dynamic Range Extension to JPEG

Greg Ward, BrightSide Technologies, Albany, California, USA
Maryann Simmons, Walt Disney Feature Animation, California, USA

Abstract

The transition from traditional 24-bit RGB to high dynamic range (HDR) images is hindered by excessively large file formats with no backwards compatibility. In this paper, we demonstrate a simple approach to HDR encoding that parallels the evolution of color television from its grayscale beginnings. A tone-mapped version of each HDR original is accompanied by restorative information carried in a subband of a standard output-referred image. This subband contains a compressed ratio image*, which when multiplied by the tone-mapped foreground, recovers the HDR original. The tone-mapped image data is also compressed, and the composite is delivered in a standard JPEG wrapper. To naïve software, the image looks like any other, and displays as a tone-mapped version of the original. To HDR-enabled software, the foreground image is merely a tone-mapping suggestion, as the original pixel data are available by decoding the information in the subband. Our method further extends the color range to encompass the visible gamut, enabling a new generation of display devices that are just beginning to enter the market.*

Introduction

Visible light in the real world covers a vast dynamic range. Humans are capable of simultaneously perceiving over 4 orders of magnitude (1:10000 contrast ratio), and can adapt their sensitivity up and down another 6 orders. Conventional digital images, such as 24-bit *sRGB*,[1] hold less than 2 useful orders of magnitude. Such formats are called *output-referred* encodings because they are tailored to what can be displayed on a common CRT monitor – colors outside this limited gamut are not represented. A *scene-referred* standard is designed instead to represent colors and intensities that are visible in the real world, and though they may not be rendered faithfully on today's output devices, they may be visible on displays in the near future.[2] Such image representations are referred to as *high dynamic range* (HDR) formats, and a few alternatives have been introduced over the past 15 years, mostly by the graphics research and special effects communities.

Unfortunately, none of the existing HDR formats supports lossy compression, and only one comes in a conventional image wrapper. These formats yield prohibitively large images that can only be viewed and manipulated by specialized software. Commercial hardware and software developers have been slow to embrace scene-referred standards due to their demands on image capture, storage, and use. Some digital camera manufacturers attempt to address the desire for greater exposure control during processing with their own proprietary RAW formats, but these camera-specific encodings fail in terms of image archiving and third-party support.

What we really need for HDR digital imaging is a compact representation that looks and displays like an output-referred JPEG, but holds the extra information needed to enable it as a scene-referred standard. The next generation of HDR cameras will then be able to write to this format without fear that the software on the receiving end won't know what to do with it. Conventional image manipulation and display software will see only the tone-mapped version of the image, gaining some benefit from the HDR capture due to its better exposure. HDR-enabled software will have full access to the original dynamic range recorded by the camera, permitting large exposure shifts and contrast manipulation during image editing in an extended color gamut. The availability of an efficient, backwards-compatible HDR image standard will provide a smooth upgrade path for manufacturers and consumers alike.

Background

High dynamic range imaging goes back many years. A few early researchers in image processing advocated the use of logarithmic encodings of intensity (e.g., Ref. [3]), but it was global illumination that brought us the first standard. A space-efficient format for HDR images was introduced in 1989 as part of the *Radiance* rendering system.[4,5] However, the *Radiance* RGBE format was not widely used until HDR photography and image-based lighting were developed by Debevec.[6,7] About the same time, Ward Larson[8] introduced the LogLuv alternative to RGBE and distributed it as part of Leffler's public TIFF library.[9] The LogLuv format has the advantage of covering the full visible gamut in a more compact representation, whereas RGBE is restricted to positive primary values. A few graphics researchers adopted the LogLuv format, but most continued to use RGBE (or its extended gamut variant XYZE), until Industrial Light and Magic made their EXR format available to the community in 2002.[10] The OpenEXR library uses the same basic 16-bit/channel floating-point data type as modern graphics cards, and is poised to be the new favorite in the special effects industry. Other standards have also been proposed or are in the works, but they all have limited dynamic range relative to their size (e.g., Ref. [11]). None of these standards is backwards compatible with existing software.

The current state of affairs in HDR imaging parallels the development of color television after the adoption of black and white broadcast. A large installed base must be accommodated as well as an existing standard for transmission. The NTSC solution introduced a subband to the signal that encoded the additional chroma information without interfering with the original black and white signal.[12] We propose a similar solution in the context of HDR imagery, with similar benefits.

As in the case of black and white television, we have an existing, de facto standard for digital images: output-referred JPEG. JPEG has a number of advantages that are difficult to beat. The standard is unambiguous and universally supported. Software libraries are free and widely available. Encoding and decoding is fast and efficient, and display is straightforward. Compression performance for average quality images is competitive with more recent advances, and every digital camera writes to this format. Clearly, we will be living with JPEG images for many years to come. Our chances of large scale adoption increase dramatically if we can maintain backward compatibility to this standard.

Our general approach is to introduce a subband that accompanies a tone-mapped version of the original HDR image. This subband is compressed to fit in a metadata tag that will be ignored by naïve applications, but can be used to extract the HDR original in enabled software. We utilize JPEG/JFIF as our standard wrapper in this implementation, but our technique is compatible with any format that provides client-defined tags (e.g., TIFF, PNG, etc.).

The Method section of our paper describes the ideas behind HDR subband encoding, followed by specific details in the Implementation section. The Results section presents some examples and compression statistics. We end with our conclusions and future directions.

Method

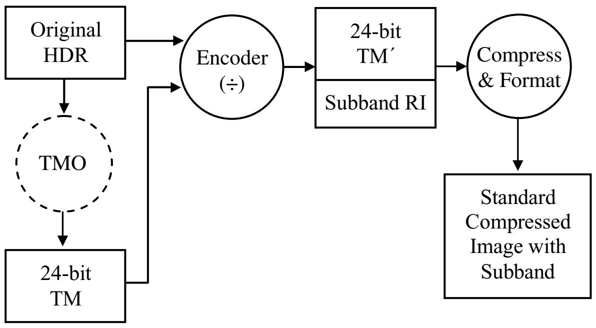

Figure 1. High level HDR subband encoding pipeline.

Figure 1 shows an overview of our HDR encoding pipeline. We start with two versions of our image: a scene-referred HDR image, and an output-referred, tone-mapped version. If we like, we can generate this tone-mapped version ourselves, but in general this is a separable problem, and our implementation is compatible with multiple operators. The encoding stage takes these two inputs and produces a composite, consisting of a potentially modified version of the original tone-mapped image, and a subband ratio image that contains enough information to reproduce a close facsimile of the HDR original. The next stage compresses this information, offering the tone-mapped version as the JPEG base image, and storing the subband as metadata in a standard JFIF wrapper.

Figure 2 shows the two possible decoding paths. The high road decompresses both the tone-mapped foreground image and the subband, delivering them to the decoder to recover the HDR pixels. Naïve applications follow the low road, ignoring the subband in the metadata and reproducing only the tone-mapped foreground image.

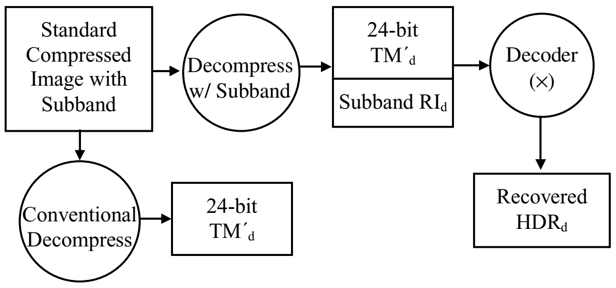

Figure 2. Alternate decoding paths for compressed composite.

In the simplest incarnation of this method, the encoder divides the HDR pixels by the tone-mapped luminances, producing an 8-bit ratio image that is stored as metadata in a standard JPEG compressed image. Decoding follows decompression with a multiplication step to recover the original HDR pixels. Unfortunately, the simplest approach fails for pixels that are mapped to black or white by the tone-mapping operator (TMO), and the standard clamping of RGB values can result in a substantial loss of color saturation at the top end. These are two important issues we must address with our technique, and these are discussed in the following subsections.

A similar approach to JPEG gamut extension has been implemented in Kodak's ERI system.[13] In the ERI format, a *residual image* rather than a ratio image is recorded in a subband. This residual is defined as the arithmetic difference between an input ERIMM RGB color space and the recorded sRGB foreground image, reduced to an 8-bit range. Unfortunately, the 12 bits present in the original data cannot be recovered by adding two 8-bit channels together. Kodak's reconstructed image gains at most two f-stops in dynamic range over sRGB (about 2.2 orders of magnitude total), rather than the four orders of magnitude we need for true HDR imagery.

The Ratio Image

JPEG/JFIF provides a limited set of metadata storage channels, called the "application markers." Sixteen application markers exist in the JFIF specification, three of which are spoken for. A single marker holds a maximum of 64 Kbytes of data, regardless of the image dimensions. This 64K limit can be circumvented by storing multiple markers reusing the same identifier, and this is what we have done in our implementation. The foreground signal in our encoding is a tone-mapped version of the original HDR image. This output-referred image is stored in the usual 8×8, 8-bit blocks using JPEG's lossy DCT compression.[14] The subband encodes the information needed to restore the HDR original from this compressed version.

Let us assume that our selected tone-mapping operator possesses the following properties:

- The original HDR input is mapped smoothly into a 24-bit output domain, with no components being rudely clamped at 0 or 255.

- Hue is maintained at each pixel, and mild saturation changes can be described by an invertible function of input and output value.

Most tone-mapping operators for HDR images have the first property as their goal, so this should not be a problem. If it is, we can override the operator by locally replacing each clamped pixel with a lighter or darker value that fits the range. Similarly, we can enforce the second property by performing our own color desaturation, using the tone-mapping operator as a guide only for luminance changes. Most tone-mapping operators address color in a very simple way, if they consider it at all. Exceptions are found in operators that simulate human vision (e.g., Ref. [15]), whose support is outside the scope of this encoding.

Given these restrictions, a ratio image may be computed by dividing the HDR original luminance at each pixel by the tone-mapped output luminance:

$$RI(x,y) = \frac{L(HDR(x,y))}{L(TM(x,y))} \quad (1)$$

The ratio image is log-encoded and compressed as an 8-bit greyscale image stored in our subband along with the saturation formula described in the following subsection. The range of the ratio image is determined beforehand in order to optimize the subband encoding's precision. The tone-mapped (and desaturated) version is then encoded as the foreground image. During reconstruction, color is restored using the recorded saturation parameters. The ratio image is then remultiplied pixel by pixel to recover the original HDR image. An example tone-mapped foreground image and its associated subband is shown in Figure 3.

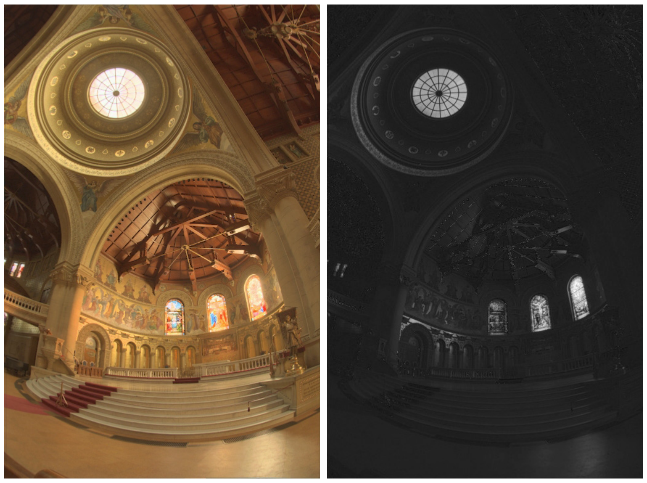
Figure 3. Memorial Church shown tone-mapped with Reinhard's global operator (left) along with the log-encoded ratio image needed for HDR recovery (right).

Color Saturation and Gamut

To preserve colors during transcoding through the output-referred space of our foreground image, we apply two separate but complimentary techniques. The first technique makes use of normally unused YCC code values to extend the working gamut. The second technique desaturates colors prior to encoding, then applies resaturation during decoding (i.e., gamut companding).

In the first technique, we take advantage of the fact that the standard 24-bit YCbCr space employed in JPEG is larger than the sRGB space it encodes. As sRGB nears its white value (255,255,255), its gamut triangle gets vanishingly small. In comparison, the YCC space has a generous gamut as it approaches its white (255,128,128), because the chroma values can take on any value from 0-255, 128 being neutral. By ascribing meaning to these otherwise unused YCC code values, it is possible to extend the effective gamut of a YCC image to include bright, saturated colors that would otherwise be lost. This strategy has been employed successfully by Canon and Sony to preserve otherwise out-of-gamut colors, and there is currently an effort underway to standardize such an extension by IEC TC100. By itself, such an extension to YCC does not provide additional dynamic range because the Y value is still clamped at 255, but it is a perfect complement to our greyscale subband, since it provides the extra saturation we need near the top of the range. Otherwise, we would have to introduce an unnatural darkening to regions of the foreground image where the bright values happened to be colorful, as in a sunset.

The YCC extension we have implemented precisely matches the standard sRGB to YCbCr mapping where it is defined, then logically extends past the boundaries to encompass additional colors. Because we expect linear floating point RGB input values, we define our mapping to and from linear RGB, bypassing sRGB while maintaining consistency with it. We assume that the RGB color space uses the standard primaries and white point given in Table 1. However, we permit our input channels to take on values outside the 0-1 range, mirroring our gamma lookup to include negative values as defined below:

$$R' = \begin{matrix} 1.055R^{0.42} - 0.055 & \text{if } R > 0.0031308 \\ 12.92R & \text{if } |R| \le 0.0031308 \\ -1.055(-R)^{0.42} + 0.055 & \text{if } R < -0.0031308 \end{matrix} \quad (2)$$

This equation is then repeated for G' and B', which are combined with the following standard transform to yield our mapping from RGB to YCC:

$$\begin{aligned} Y &= (0.299R' + 0.587G' + 0.114B') \times 256 \\ Cb &= (-0.169R' - 0.331G' + 0.5B' + 0.5) \times 256 \\ Cr &= (0.5R' - 0.419G' - 0.813B' + 0.5) \times 256 \end{aligned} \quad (3)$$

Values outside the allowed 8-bit range (0-255) must of course be truncated, but even so this mapping provides a much larger gamut near white. To extend the gamut even further, we apply a second technique, *gamut companding*.

Table 1: sRGB Primary Chromaticities

	R	*G*	*B*	*W*
x	0.640	0.300	0.150	0.3127
y	0.330	0.600	0.060	0.3290

To compress the gamut during encoding, we can apply a global desaturation to the image, pulling all colors in towards gray by an amount that is guaranteed to contain the entire visible gamut in our extended YCC range. This may also be beneficial to the appearance of the foreground image, as dynamic range compression tends to leave the colors with an oversaturated appearance. The gamut is then expanded during HDR decoding, rendering the colors back into their original saturation. We employ the following definition of color saturation based on the linear values:

$$S \equiv 1 - \min(R,G,B)/Y \tag{4}$$

Y in the formula above is the luminance of red, green and blue taken together (0.213*R + 0.715*G + 0.072*B). Note that we expect saturation to be greater than 1 for negative primaries. Zero saturation implies a neutral value, which is passed unchanged. Non-neutral values are desaturated with a two-parameter formula to obtain our compressed color saturation:

$$S_c = \alpha \cdot S^{\beta} \tag{5}$$

The α parameter controls how much saturation we wish to keep in the encoded colors, and is generally ≤ 1. The β parameter controls the color "contrast," and may be greater or less than 1 depending on the desired effect. This modified saturation is used with the original saturation from Eq. (4) to determine the encoded primary values. Below is the formula for the desaturated red channel:

$$R_c = 1 - \frac{S_c}{S} \cdot Y + \frac{S_c}{S} \cdot R \tag{6}$$

This equation is repeated for the green and blue channels. Note that *Y* does not change under this transformation, and the primary that was smallest before is the smallest after. The R_c in Eq. (6) then becomes the input *R* in Eq. (2) for the YCC mapping, and similarly for G_c and B_c.

Resaturating the encoded color to get back the original pixel is done by inverting these equations. If the smallest primary value were blue for example, this inverse transformation would yield:

$$B = Y - Y \cdot {\frac{Y - B_c}{\alpha Y}}^{1/\beta} \tag{7}$$

The red and green channels would then be determined by:

$$R = Y - \frac{(Y - R_c)}{\alpha} {1 - \frac{B}{Y}}^{1-\beta}$$
$$G = Y - \frac{(Y - G_c)}{\alpha} {1 - \frac{B}{Y}}^{1-\beta} \tag{8}$$

If either red or green were the minimum primaries, these same equations hold, with the color values substituted accordingly.

Gamut companding applies to our linear RGB values, prior to YCC encoding during compression and following YCC decoding during expansion. The subband contains no color information, but it, too, is compressed using the same JPEG DCT method employed for the color foreground image. The difference is that one log-encoded channel is present instead of three gamma-encoded channels.

Subband Compression

Downsampling the ratio image can greatly reduce the size of the subband without serious consequences to the recovered HDR image's appearance. As in the perception of color, our eye has a limited ability to register large, high frequency changes in luminance. This is due primarily to optical scattering in the eye, which has also been exploited to make efficient HDR displays with different resolution modulators.[2] Note that we can and do see small, high frequency luminance changes, but these can be adequately represented in the foreground image. By trading resolution with the foreground, we may obtain much better compression using a downsampled subband. Details of how this is done and the trade-offs involved are described in Ref. [16].

To summarize, we have implemented two alternative methods for subband downsampling: a *precorrection* method and a *postcorrection* method. The precorrection method starts by downsampling and then upsampling the ratio image in order to predict what the decompressor will see on the receiving end. Equation (1) is then rearranged so that the resampled ratio image, RI_d, is divided into the original HDR luminance values to obtain a pre-corrected, tone-mapped *Y* channel:

$$TM' = \frac{HDR}{RI_d} \tag{9}$$

By construction, this pre-corrected foreground image will then reproduce the HDR original when it is multiplied by the upsampled ratio image during decompression. The only visible effect is an increase in sharpness at high contrast boundaries in the foreground image. This sharpening will then cancel out in the recovered HDR image. Precorrection is therefore, the most appropriate method when the fidelity of the decoded HDR image is paramount.

The alternative postcorrection method is more appropriate when the appearance of the output-referred image must not be compromised. In this method, we synthesize the high frequencies that we lost to downsampling of the ratio image using our foreground image as a guide. This is a simple application of resolution enhancement via example learning. Much more sophisticated techniques have been developed by the vision community [e.g., Ref. [17]. We refer the reader to our previous paper for details and examples of this method.[16]

Implementation

We have implemented a fully functional library for reading and writing our compressed, high dynamic range format, which we call JPEG-HDR. As described above, it employs a standard JPEG/JFIF wrapper to hold the foreground image, and encodes a subband containing a compressed ratio image in one or more application 11 markers.[18] In addition to this JPEG-compressed greyscale image, the subband contains the lower and upper ratios corresponding to 0 and 255, the correction method used if the image was downsampled, the saturation parameters α and β, and a calibration factor for recovering absolute luminance values where available.

Since one of the key goals of this format is backwards compatibility, we based our implementation on the existing, widely-used and public IJG (Independent JPEG group) library written by Thomas Lane.[19] Most imaging applications either use this library or some modified version of it, so it serves as a common reference for code development. By matching the *libjpeg* call structure and leveraging its routines, we hope to minimize the coding effort required to support our JPEG-HDR extension. In fact, the code we have written links to an unmodified version of the IJG library, and works interchangeably on Windows, Linux, and Apple platforms.

The code changes required to support the reading and writing of HDR data are minimally invasive, requiring in most cases nothing more than the substitution of a few calls and data structures. For example, in place of the standard *libjpeg* jpeg_create_decompress() call to initialize a reader structure, an HDR-enabled application calls jpeghdr_create_decompress(), which initializes both a standard JPEG reader structure and the extended structure containing it. The subsequent call to jpeghdr_read_header() then returns either JPEG_HEADER_OK in the case of a standard JPEG, or JPEG_HEADER_HDR in the case of a JPEG-HDR image. If the image is a standard JPEG, then the application proceeds as before, calling the normal *libjpeg* routines. If the image is a JPEG-HDR, however, the application uses our replacement routines to recover floating-point RGB scanlines rather than the more usual sRGB data. (For convenience, we also provide a means to access the tone-mapped YCC or sRGB data for applications that want it.)

As a further convenience for new applications, we also provide a simplified pair of calls for reading and writing JPEG-HDR images to and from memory. The jpeghdr_load_memory() routine takes a named JPEG image file and allocates either a floating-point or 8-bit frame buffer depending on its type, and reads the image into memory. Similarly, the jpeghdr_write_memory() routine takes a floating-point frame buffer and writes it out to a JPEG-HDR image. Parameters are provided for setting the quality, correction method, and gamut compression.

Bear in mind that it is not necessary to use our extended library to read a JPEG-HDR image, as the format itself is backwards-compatible with the JPEG/JFIF standard. Any application that reads standard JPEG images will be able to read and display JPEG-HDR with no changes or recompilation. It will only see the tone-mapped foreground image in this case, but if the application is not HDR-enabled, this is the best visual representation available to it.

Tone-Mapping

As mentioned in the previous section, our framework is designed to function with multiple tone-mapping operators. We have experimented with a selection of global and local TMOs, including Reinhard's global photographic tone operator,[20] Ward Larson's global histogram adjustment,[21] Fattal's gradient operator,[22] and Durand & Dorsey's bilateral filter.[23] Of these, we found that the bilateral filter performed best with our method, followed closely by Reinhard's photographic TMO.[16]

In our library implementation, we provide a default mapping based on Reinhard's TMO, along with hooks so the calling application can substitute its own luminance mapping if desired, similar to the way applications can provide their own JPEG quantization tables. These hooks include a method for computing the log histogram of luminance, as required by most global TMOs, and a call to check whether or not a tone-mapped RGB color is inside the compressed YCC gamut of our foreground image. The latter is convenient for adjusting the tone-mapping to minimize color saturation losses in the encoded image, and is used for this purpose in the default TMO provided.

We fully expect that custom tone-mappings will be a key "value added" area for software and hardware vendors using HDR in the future, and the separation of the TMO from the decoding method is one of the key strengths of this format.

Results

JPEG-HDR has been fully integrated into Photosphere 1.3, a freeware application for building and cataloging HDR images under Mac OS X. We have converted hundreds of HDR images to the JPEG-HDR format and studied compression performance, looking for artifacts and errors, and we consider the library to be well-tested at this stage.

Figure 4 shows compression performance statistics for different quality settings on a collection of 217 natural scene captures. The JPEG-HDR image size, in bits per pixel, is plotted as a function of the quality setting. Although the quality may be set anywhere from 0-100, the appearance degrades rapidly below a setting of 60, so we consider the range here to be representative. There was only a small difference in total file size between the precorrection and postcorrection methods, precorrection being larger because it introduces additional high frequency information into the foreground image. (There is no subband downsampling at quality levels above 95, which is why the Q=99 points are the same.)

The JPEG-HDR format achieved average bits/pixel rates from 0.6 to 3.75 for corresponding quality settings from 57-99%. For comparison, the Radiance RGBE format[4] uses on average 26 bits/pixel for this image set, and OpenEXR[24] uses 28 bits/pixel. The best compression performance we measured for a lossless format was LogLuv TIFF,[25] which averaged 21.5 bits/pixel for this data set. Thus, relative to these lossless formats, JPEG-HDR offers additional compression of between 6:1 (84%) and 40:1 (97%) over lossless methods, depending on quality.

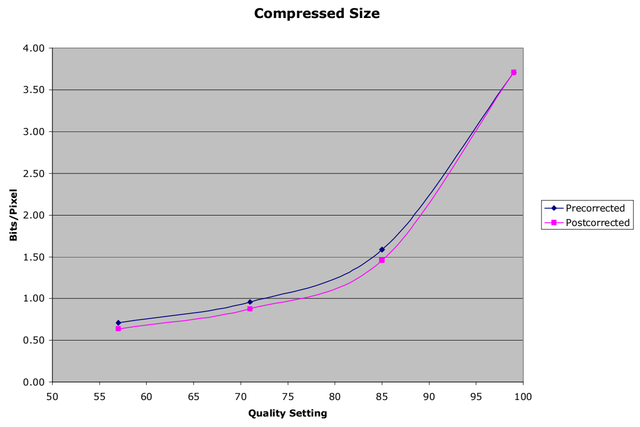

Figure 4. JPEG-HDR compressed image size.

Figure 5 shows the fraction of the saved file taken up by the ratio image subband. At lower quality levels, the ratio image is downsampled at a higher rate, which leads to a smaller percentage of the total file size. On average, the ratio image occupies less than 1/4 of the total file size, though this percentage can vary quite a bit from one image to the next. In our image captures, the compressed subband took between 16% and 27% of the file at Q=71, and between 24% and 37% at Q=85. The actual space taken up by the subband is about the same for the precorrection and postcorrection cases. It ends up being a larger fraction for the postcorrection case simply because the foreground image ends up being smaller, as we just explained.

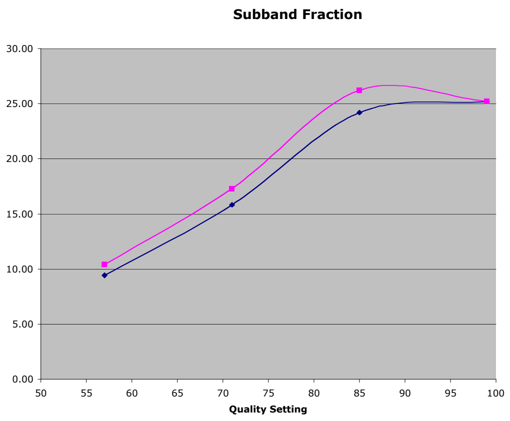

Figure 5. JPEG-HDR subband size.

Figure 6 shows another view of the Memorial church in false color, indicating a luminance range that spans over 5 orders of magnitude. Because the artifacts are most visible in the darkest regions of our image, we have magnified the right-hand alcove ceiling for the our visual quality comparison and normalized the exposure with a linear scale that is a factor of 8000 (13 stops) above the saturation level for the brightest region. The results shown in Figure 7 at are nearly perfect at Q=99, even in the deepest shadows. As we decrease our quality setting (and our bitrate), we see that the image degrades in a controlled way. At the lowest quality we tested, Q=57, block artifacts are visible at the ceiling's apex, and ringing is evident near the light fixtures.

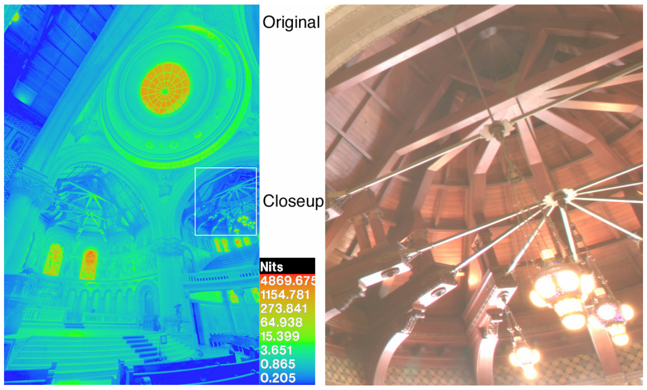

Figure 6. A close-up of one of the darker sections of the Memorial church, shown in its original, uncompressed form.

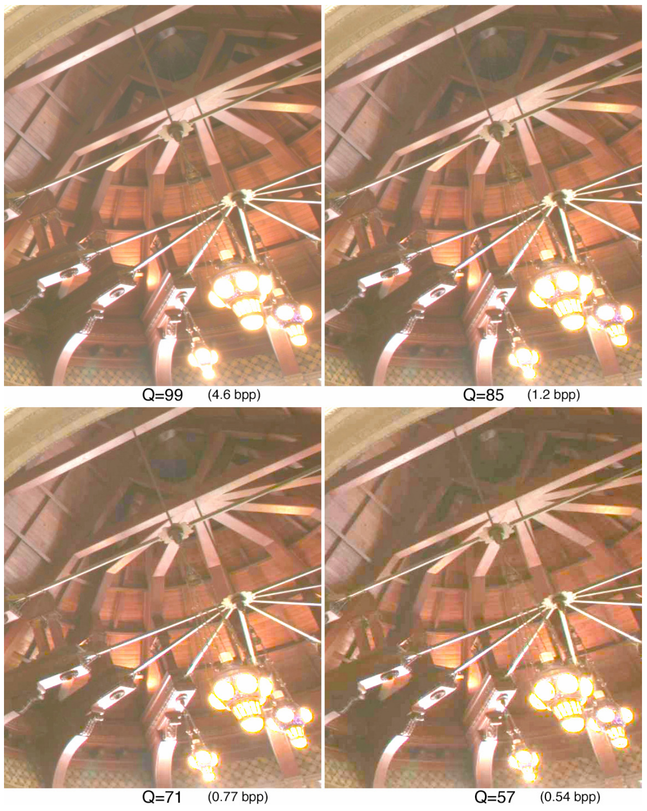

Figure 7. Quality of recovered image at different compression settings using subband precorrection method and Reinhard's global TMO (bits/pixel in parentheses).

To test our algorithm more systematically, we used Daly's Visible Differences Predictor[24] to evaluate image fidelity before and after JPEG-HDR compression. At a quality setting of 90, we found fewer than 2.5% of the HDR image pixels (on average) were visibly different after decompression using Reinhard's global TMO and our subband precorrection method. This increased to

4.7% with the postcorrection method. At the maximum quality setting, fewer than 0.1% of the pixels were visibly different in each image.[16]

Conclusions and Future Directions

Although one could implement a lossy high dynamic-range format as an extension to JPEG 2000, it would not have the benefit of backwards compatibility with existing hardware and software. Does that really matter? Certainly, for HDR-specific applications, backwards compatibility is of little import. However, more traditional imaging applications are bound to be slow in adopting HDR because of the additional burden it places on tone-mapping for viewing and output. Consider how long it has taken for ICC profiles to see widespread use. The ICC is just now coming to terms with scene-referred standards and what they mean to the imaging pipeline. The more difficult it is for software vendors to support a new color technology, the longer it will take to reach the mainstream. Backwards compatibility offers us a shortcut, a way to bring HDR to the masses before they come to us, demanding it. We will all get there, eventually. In the meantime, JPEG-HDR allows the advance guard to share scene-referred images without bringing down the network.

High dynamic-range imaging has enormous potential for digital photography, which has been suffering since its inception from the mistaken but convenient notion that a camera is somehow like a scanner, and the same color strategy that works for prepress can work for photography. It cannot and it does not. There is no white in the real world. There is only brighter and darker, redder and bluer. "Scene-referred color" is unbounded and unreferenced. We may choose a white point, but in practice our choice is based on scene statistics and heuristics rather than any known or measurable illuminant. Similarly, we should never attempt to define reflectance values in photography. There is no such thing. If we set a gray card in the scene and call that 20%, it is relative to the light falling on it in that position and orientation at that moment. If someone's shadow passes over our reference, its brightness may drop by a factor of 100, and yet we still call that 20%? Our notions and our imaging techniques must change to fit with the reality, not the other way around. Photography measures light, not reflectance. Digital images should record light, and for this we need an HDR format. A backwards-compatible, scene-referred extension to JPEG allows us to live with our mistake *while* we correct it.

The most obvious and important future work in this area is an HDR extension to MPEG. Mantiuk et al. suggested one such extension, though their solution is not backwards-compatible.[25] Backwards compatibility is even more critical to video applications, as the installed base of DVD players is not going away anytime soon. A version of MPEG that does not play on existing hardware is likely to be confined to the laboratory until the next major video format revolution, which is at least 10 years away by most predictions.

Fortunately, the extension of the subband method to MPEG is reasonably straightforward. To achieve low bitrates, a subband needs to be compressed across multiple frames, and this is most easily accomplished as a sub-video within the MPEG main stream. Just as we were able to smuggle our ratio image in a JPEG marker, we can smuggle a *ratio video* as MPEG metadata. In specially equipped players, this subband will be decoded simultaneously with the main MPEG stream and recombined in a multiplication step for HDR viewing on an appropriate display.[2] If no HDR player or display is available, the foreground video's tone-mapping will already provide an appropriate color space for low dynamic-range output. As in still imagery, this approach allows the gradual introduction of HDR to the video market, a market we know will not accept any other type of introduction.

Software Availability

Photosphere, an HDR image builder and browser for Mac OS X, reads, writes, and converts JPEG-HDR and is available for free download from www.anyhere.com. Additional software, including format converters and a non-commercial JPEG-HDR library for Windows, Apple, and Linux platforms are available on the DVD-ROM that accompanies.[26] For commercial inquiries, please visit the BrightSide Technologies website at www.brightsidetech.com/products/process.php

References

1. Stokes, M., Anderson, M., Chandrasekar, S., and Motta, R. 1996. Standard Default Color Space for the Internet. www.w3.org/Graphics/Color/sRGB.
2. Seetzen, H., Heidrich, W., Stuezlinger, W., Ward, G., Whitehead, L., Trentacoste, M., Ghosh, A., Vorozcovs, A. 2004. "High Dynamic Range Display Systems," *ACM Trans. Graph.* (special issue SIGGRAPH 2004).
3. Jourlin, M., Pinoli, J-C., 1988. A model for logarithmic image processing, *Journal of Microscopy*, 149(1), pp. 21-35.
4. Ward, G. 1991. Real Pixels. In *Graphics Gems II*, edited by James Arvo, Academic Press, 80-83.
5. Ward, G. 1994. The RADIANCE Lighting Simulation and Rendering System. , In *Proceedings of ACM SIGGRAPH 1994*, 459-472.
6. Debevec, P., and Malik, J. 1997. Recovering High Dynamic Range Radiance Maps from Photographs. In *Proceedings of ACM SIGGRAPH 1997*, 369-378.
7. Debevec, P. 1998. Rendering Synthetic Objects into Real Scenes: Bridging Traditional and Image-Based Graphics with Global Illumination and High Dynamic Range Photography. In *Proceedings of ACM SIGGRAPH 1998*, 189-198.
8. Ward Larson, G. 1998. Overcoming Gamut and Dynamic Range Limitations in Digital Images. *Proc. of IS&T 6th Color Imaging Conf.*
9. Leffler, S., Warmerdam, F., Kiselev, A., 1999. *libTIFF*. remotesensing.org/libtiff.
10. Kainz, F., Bogart, R., Hess, D., Schneider, P., Anderson, B., 2002. *OpenEXR*. www.openexr.org/.
11. IEC. 2003. 61966-2-2. Extended RGB colour space – scRGB, *Multimedia systems and equipment – Colour measurement and management* – Part 2-2: Colour management.
12. Jolliffe, C.B., 1950. Answers to Questions about Color Television. members.aol.com/ajaynejr/rca2.htm.
13. Spaulding, K., Woolfe, G., & Joshi, R. 2003. "Using a Residual Image to Extend the Color Gamut and Dynamic Range of an sRGB Image," white paper posted on the Kodak website.
14. Wallace, G. 1991. The JPEG Still Picture Compression Standard. *Communications of the ACM*, 34, 4, 30-44.
15. Pattanaik, S., Ferwerda, J., Fairchild, M., and Greenberg, D. 1998. A Multiscale Model of Adaptation and Spatial Vision for Realistic Image Display, In *Proceedings of ACM SIGGRAPH 1998*, 287-298.

16. Ward, G. & Simmons, M. 2004. “Subband Encoding of High Dynamic Range Imagery,” *First Symposium on Applied Perception in Graphics and Visualization* (APGV).
17. Baker, S & Kanade, T. 2002. Limits on super-resolution and how to break them. IEEE Transactions on Pattern Analysis and Machine Intelligence., 24(9):1167-1183, September. 2002.
18. Spinal Tap, This i*s*. 1984. Dir. Rob Reiner. MGM Home Entertainment.
19. Independent JPEG Group. 1998. www.ijg.org/
20. Reinhard, E., Stark, M., Shirley, P., and Ferwerda, J. 2002. Photographic Tone Reproduction for Digital Images. *ACM Transactions on Graphics*, 21,3, 267-276.
21. Ward Larson, G., Rushmeier, H., and Piatko, C. 1997. A Visibility Matching Tone Reproduction Operator for High Dynamic Range Scenes. *IEEE Trans. on Visualization and Computer Graphics*, 3, 4.
22. Fattal, R., Lischinski, D., and Werman, M. 2002. Gradient Domain High Dynamic Range Compression. *ACM Transactions on Graphics*, 21, 3, 257-266.
23. Durand, F., and Dorsey, J. 2002. Fast Bilateral Filtering for the Display of High-Dynamic Range Images. *ACM Transactions on Graphics*, 21, 3, 249-256.
24. Daly, S. 1993. The Visible Differences Predictor: An Algorithm for the Assessment of Image Fidelity. In *Digital Images and Human Vision*, A.B. Watson, editor, MIT Press, Cambridge, Massachusetts.
25. Mantiuk, R., Krawczyk, G., Myszkowski, K., Seidel, H-P. 2004. “Perception-motivated High Dynamic Range Video Encoding,” *SIGGRAPH 2004*.
26. Reinhard, E., Ward, G., Pattanaik, S., Debevec, P. 2005. *High Dynamic Range Imaging: Acquisition, Display, and Image-based Lighting*, Morgan Kaufmann Publishers, San Francisco.