High-Dynamic-Range Still-Image Encoding in JPEG 2000

Ruifeng Xu, Sumanta N. Pattanaik, and Charles E. Hughes
University of Central Florida

The raw size of a high-dynamic-range (HDR) image brings about problems in storage and transmission. Many bytes are wasted in data redundancy and perceptually unimportant information. To address this problem, researchers have proposed some preliminary algorithms to compress the data, like RGBE/XYZE, OpenEXR, LogLuv, and so on.[1] These algorithms mostly use lossless compression strategies, so they are not capable of providing significant compression. However, lossy compression alternatives have started becoming available, such as Ward and Simmons'[2] subband encoding and Mantiuk et al.'s[3] perception-motivated encoding (see the "Related Work" sidebar).

HDR images can have a dynamic range of more than four orders of magnitude while conventional 8-bit images retain only two orders of magnitude of the dynamic range. This distinction between an HDR image and a conventional image leads to difficulties in using most existing image compressors. JPEG 2000 supports up to 16-bit integer data, so it can already provide image compression for most HDR images.

In this article we propose a JPEG 2000-based lossy image compression scheme for HDR images of all dynamic ranges. We show how to fit HDR encoding into a JPEG 2000 encoder to meet the HDR encoding requirement. To achieve the goal of minimum error in the logarithm domain, we map the logarithm of each pixel value into integer values and then send the results to a JPEG 2000 encoder. Our approach is basically a wavelet-based HDR still-image encoding method.

The authors use JPEG 2000 to compress high-dynamic-range still images anywhere in the spectrum—from very low bit rate to visually lossless.

HDR image compression requirements

An HDR image records the photometric measurements of each pixel and has no restrictions based on any final display and/or viewing condition. This display and viewing independence generally results in a high dynamic range. Because the human visual system has a near-logarithmic response to a wide dynam-

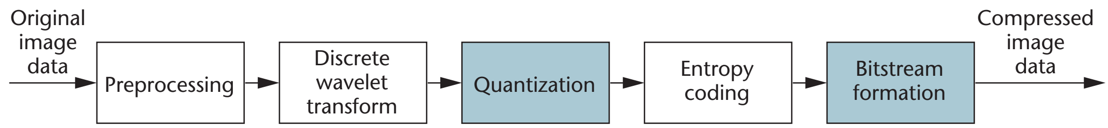
Related Work

Mantiuk et al.[1] proposed a hybrid encoding method, which applies compression to the high-dynamic-range (HDR) video in two stages, first to the luminance channel and then to the resulting image in its frequency domain. The method first quantizes the luminance channel of the HDR video using a nonlinear function, which distributes the quantization errors to match the luminance threshold of the human visual system in changing adaptation levels. It then sends the result from the first step to an extended discrete cosine transform (DCT)-based MPEG-4 video encoder for further data compression in the frequency domain. To suppress the visual artifacts around the edges of sharp luminance changes, it supplements the MPEG-4 video encoder by extracting the edges around sharp luminance changes from the DCT blocks and compressing them separately using the run length encoding (RLE) method.

Ward and Simmons describe a lossy HDR image encoding method.[2] Their method separates the HDR image into an 8-bit low-dynamic-range (LDR) image, and a ratio image. The LDR image is computed by applying a tone-mapping operator to the original HDR image. The ratio image is the ratio of the luminance images of the original and the tone-mapped version. Both components of the HDR image are compressed using JPEG separately. The compressed ratio image is stored in the subband channels of the compressed LDR image. It's also a DCT-based encoding scheme. The method suffers from blocking artifacts of DCT encoding, and from the loss of chrominance information in the bright regions of an HDR image.

References

1. R. Mantiuk et al., "Perception-Motivated High-Dynamic-Range Video Encoding," *ACM Trans. Graphics*, vol. 23, no. 3, 2004, pp. 733-741.
2. G. Ward and M. Simmons, "Subband Encoding of High Dynamic Range Imagery," *Proc. 1st Symp. Applied Perception in Graphics and Visualization* (APGV), ACM Press, 2004, pp. 83-90.



1 JPEG 2000's fundamental building blocks.

ic range of illumination, we aim to achieve a minimum error in the logarithm domain. This is the main goal we seek in HDR image encoding. The developers of some HDR pixel formats, like RGBE/XYZE and LogLuv, also pursue this same encoding goal.[4] The key here is that the logarithmic response curve is an approximation to the human visual system's response to a wide range of luminances and is now a part of HDR image encoding applications because of its simplicity and effectiveness.[4] A closer but more complex approximation would use a logarithm function under photopic conditions (bright light), as well as a nonlogarithmic function under mesopic (medium light) and scotopic (dim light) conditions. The response curve reported by Stevens and Stevens[5] uses a three-parameter power function to model perceived brightness to luminance under various conditions. The display related issues, like gamma correction, are considered during the display time together with tone mapping. Separating HDR image display from HDR image encoding allows for a device-independent appearance across various displays.

JPEG 2000's compression scheme

The compression scheme in JPEG 2000[6] first transforms the raw image data into the wavelet domain and quantizes the wavelet coefficients. The scheme encodes the quantized coefficients using adaptive arithmetic coding. The final compressed data stream is formed through a rate-distortion optimization operation to meet bit-rate requirements. Figure 1 shows the whole process.

The information loss happens in the two stages colored in blue in Figure 1. The pixel bit depth is first

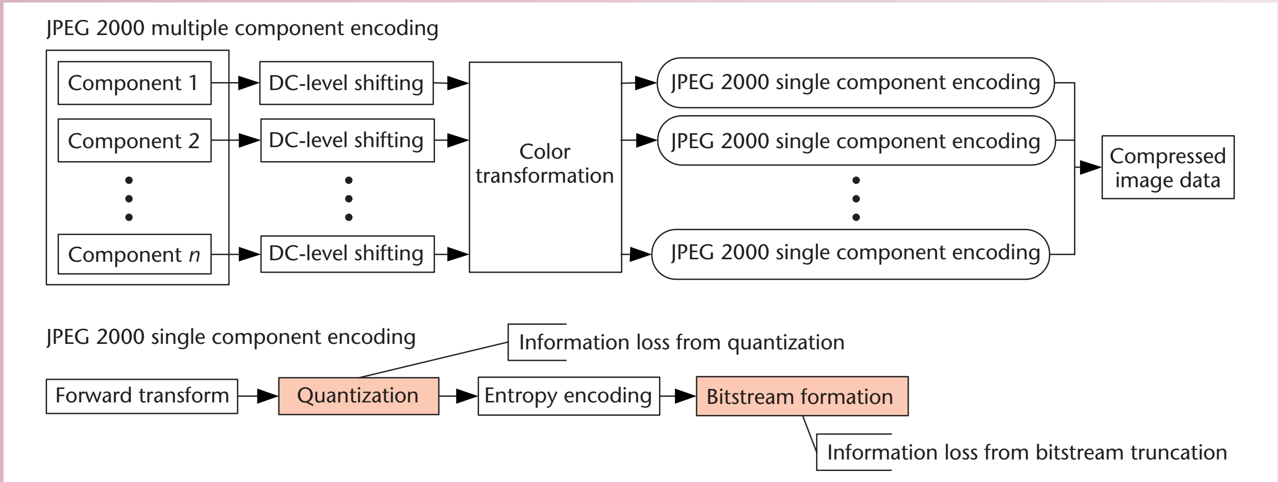
Overview of JPEG 2000

Figure A shows the various steps in the JPEG 2000 compression scheme. The inputs to JPEG 2000 encoding are multicomponent images with up to 16-bit integers (signed or unsigned) per component. Unsigned integer components undergo direct current (DC)-level shifting. The first three components after DC-level shifting might go through color transformation for color decorrelation. Each component is then encoded independently.

In the JPEG 2000 single component encoding step, the component is forward transformed into wavelet coefficients, which are then quantized for entropy encoding. The result from entropy encoding is finally truncated to the desired bit rate in the bitstream formation step.

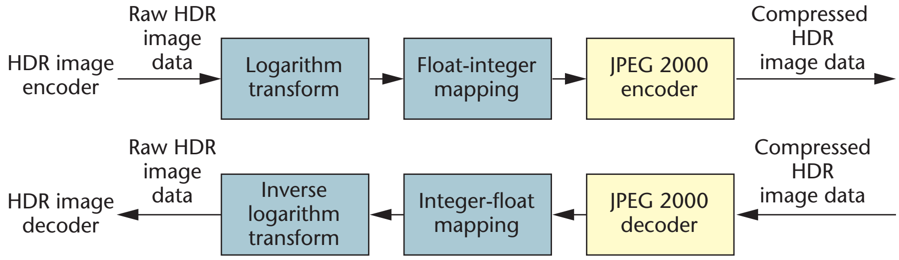
Depending on the desired mode of compression (lossy or lossless), JPEG 2000 chooses between two different color transformation matrices (reversible color transform or irreversible color transform) in the color transformation step and between two different wavelets (5/3 or 9/3) in the forward transformation step.

Two of the steps shown in Figure A might introduce information loss: quantization and bitstream formation. In the lossless compression mode, these two steps are disabled. Although the JPEG 2000 standard specification leaves the quantization coefficient's design as an implementation-specific issue, it recommends two ways to apply the human visual perception properties associated with visual frequencies: fixed visual weighting for images with a fixed viewing condition and visual progressive weighting for changing viewing conditions (see Annex J.8 of the JPEG 2000 standard[6]). To meet the desired bit rate, the bitstream formation step results in information loss due to bitstream truncation of the original bitstream during entropy encoding.



A Steps in the JPEG 2000 compression scheme.

reduced in the quantization step.[7] The table used for quantization is derived from the knowledge of the human visual system. The compressed data are then truncated in the bit-stream formation stage, where a minimum distortion is enforced within the bit-rate budget. For additional information, particularly information loss in the JPEG 2000 encoding process, see the sidebar "Overview of JPEG 2000."



2 **Our HDR image compression and decompression scheme.**

Our HDR image compression approach

Figure 2 shows our overall HDR compression and decompression scheme. The scheme has two basic components: pixel encoding/decoding and image encoding/decoding. The blue blocks in Figure 2 are the steps we introduce for pixel encoding. The yellow blocks in Figure 2 are the standard JPEG encoding and decoding schemes shown in Figure 1 that use our quantization steps for HDR image lossy compression.

Our scheme transforms the raw HDR image into the logarithm domain and then uniformly quantizes the data into n bits using the following equation :

$$\left[\bar{r}, \bar{g}, \bar{b}\right] = f\left(\left[r', g', b'\right] : n\right) \tag{1}$$

where

$$\left[r', g', b'\right] = \log\left(\left[r, g, b\right]\right)$$

$$f\left(x : n\right) = \left[\left(x - x_{\min}\right) / \left(x_{\max} - x_{\min}\right) \cdot \left(2^n - 1\right)\right]$$

r, g, b are the raw colors represented using three 32-bit floats in RGB color space; r', g', b' are logarithms of r, g, b respectively; and $\bar{r}, \bar{g}, \bar{b}$ are the colors represented in unsigned integers of n bits. $x_{\min}$ and $x_{\max}$ are the minimum and maximum value of each channel in the logarithm domain. We use floating-point numbers in logarithmic transformation, thus we have only trivial, if any, data loss in this transformation. The time consumed is acceptable for single HDR image encoding/decoding and can be improved using a GPU implementation.

The pixel encoding scheme plays an important role in preserving the color gamut and dynamic range of original raw HDR images. Our simple encoding scheme of mapping raw pixel values in three 32-bit floats into those in three n-bit integers (as Equation 1 shows) keeps the original color gamut and dynamic range—with the expense of introducing a coding error in the logarithm domain. Our method takes a nonnegative RGB color space whose color gamut covers the most commonly used colors. This constraint is common with most HDR images available to the computer graphics community.

We send the image in unsigned integers resulting from pixel encoding to the JPEG 2000 encoder for image compression. We enable the standard color transformation option available in the JPEG 2000 encoder to take advantage of color decorrelation (see Annex G of the JPEG standard[7]). This transforms the color linearly from logarithmic RGB space to YC_bC_r space. The color transform over logarithmic RGB operates in a nonlinear domain, which leads to luminance and chrominance mixing to some degree. We thus disable the chrominance subsampling, which depends on luminance and chrominance separation and is used in low-dynamic-range (LDR) image encoding.

Our approach then transforms the YC_bC_r image data to the wavelet domain. We quantize each subband b of the wavelet transformation using a quantization step Δb computed by Equation 2:

$$\Delta b = \sqrt{\gamma_{\max} / \gamma_b} \tag{2}$$

where γ_b is the energy weight for subband b, defined as the square of the amount of error introduced by a unit error in the transformed coefficient (see Annex E.2 of JPEG 2000[7]). $\gamma_{\max}$ is the maximum energy weight of all subbands.

This quantization scheme differs from the JPEG 2000 standard recommendation (see Annex J.8[7]). We chose it to maintain the independence of displaying and viewing conditions by removing the perception-related factor. HDR image formats are scene referred as opposed to LDR formats, which are image referred.

We then transform the quantized result into bitstreams through entropy encoding. The rate control mechanism in JPEG 2000 truncates the bitstream to the desired bit rate. We implemented the rate control via rate-distortion optimization, which must satisfy the bit rate constraint while minimizing the distortion (or coding error) of the reconstructed image in the logarithm domain. For more information, see the sidebar "Rate Control in JPEG 2000" (next page).

For decompression, we first decode the compressed HDR image data using a JPEG 2000 decoder. Then we convert the results to raw HDR image data via the inverse operation of Equation 1:

$$\left[\bar{\bar{r}}, \bar{\bar{g}}, \bar{\bar{b}}\right] = f'\left(\left[r'', g'', b''\right] : n\right) \tag{3}$$

where

$$\left[r'', g'', b''\right] = \exp\left(\left[\bar{r}, \bar{g}, \bar{b}\right]\right)$$

$$f'\left(x : y\right) = x / \left(2^y - 1\right) \cdot \left(x_{\max} - x_{\min}\right) + x_{\min}$$

The parameters $x_{\min}$ and $x_{\max}$ in Equation 3 are the same as those in Equation 1.

Rate Control in JPEG 2000

Rate control makes the compressed image meet the desired bit rate. It assures the highest image quality possible using the desired number of bytes.

The wavelet coefficients after quantization are divided into code blocks B_i, $i = 1,2 \ldots$, which are coded separately into independent bitstreams. For code block B_i, its bitstream can be truncated into discrete lengths $R_i^1, R_i^2, \ldots,$. The corresponding distortions incurred due to the truncations are $D_i^1, D_i^2, \ldots,$. JPEG 2000 generally uses the mean squared error or weighted mean squared error as a distortion metric. The possible discrete lengths and corresponding distortions are computed and stored during the entropy coding step for use in the bitstream formation step.

Given the truncation points of the separate bitstreams of all the code blocks, the overall distortion D in the final reconstructed image is $D = D_1 + D_2 + \ldots$, and the overall length R of the final bitstream is $R = R_1 + R_2 + \ldots$. Under the constraint that R is no more than the desired bit rate $R_{\max}$, the process of finding an optimal set of truncation points that minimize D is the rate-distortion optimization issue. JPEG 2000 adopts the well-known Lagrange multipliers method[1] to solve this constrained optimization issue. The bitstream formation step is to find the truncation points by using the temporal truncation and distortion information from the entropy encoding step and then form a final bitstream comprised of the truncated separate bitstreams.

Reference

1. *Information Technology, JPEG 2000 Image Coding System—Part 1: Core Coding System*, ISO/IEC 15444-1:2000, Int'l Organization for Standardization/Int'l Electrotechnical Commission, 2000.

Note, we analyze the coding error in detail to show its sources during the encoding process.

Pixel encoding precision

We leave the parameter n for the user to manually control the coding error ε_c arising from the quantization step. We can express ε_c through Equation 4:

$$\varepsilon_c = \left(x_{\max} - x_{\min}\right) / \left(2^{n+1} - 2\right) \quad (4)$$

To restrict the maximum coding error in the logarithm domain to ε, we must use n, determined using the following equation:

$$n = \left\lceil \log_2\left(\left(x_{\max} - x_{\min}\right)/\varepsilon + 2\right) - 1 \right\rceil$$

Thus for a dynamic range of 12 orders of magnitude, and a value of n equal to 16, the coding error in the logarithm domain is $12/(2^{16+1}–2)$, or 0.01 percent.

Compared to other coding methods, it's convenient to convert the coding error in the logarithm domain to a *relative error E* using Equation 5. The relative error of some coding scheme is the ratio of the difference to the smaller value of two consecutive codes in the coding scheme:

$$E = 10^{2\varepsilon_c} - 1 \quad (5)$$

Error sources

Three operations could introduce error in the encoding process: conversion of a float to an integer, coefficient quantization, and rate-distortion optimization.

The error in lossless mode is limited only to the float conversion error:

$$\varepsilon_{\text{lossless}} = \varepsilon_c$$

The error in lossy mode, $\varepsilon_{\text{lossy}}$, is the sum of the pixel encoding error ε_c, the coefficient quantization error, ε_q, and the bitstream truncation error, $\varepsilon_{r\text{-}d}$:

$$\varepsilon_{\text{lossy}} = \varepsilon_c + \varepsilon_q + \varepsilon_{r\text{-}d}$$

The coefficient quantization, ε_q, is

$$\varepsilon_q = \sum_b \Delta b \cdot \sqrt{\gamma_b}$$

where Δb is the quantization step for subband b, and $\varepsilon_{r\text{-}d}$ is defined by Equation 6:

$$\varepsilon_{r-d} = \sqrt{\sum_i D_i^*} \quad (6)$$

where D_i^* is the distortion of code block i after rate-distortion optimization (see Annex J.10 of JPEG 2000[7]).

Compression results

We implemented our HDR compression scheme as an extension to the JasPer API (C implementation of JPEG 2000 part 1).[8] The maximum n supported by JasPer is 16. Hence, we always use $n = 16$, and use the rate-distortion (R-D) optimization of JPEG 2000 to automatically reduce the compressed data to a desired bit rate. It's possible that some raw pixel values are zero. This poses a problem for converting an image into the logarithm domain. We overcome this problem by replacing those pixels' values with the minimum nonzero channel value.

Lossless mode

In Table 1 we show the comparison statistics of our lossless compression scheme with other existing lossless schemes. The dynamic ranges (logarithm to base 10) of the four images are 4.2, 5.9, 3.6, and 4.8, respectively. For all four test HDR images, our compression scheme performed poorly compared to all others. We hypothesize that this is because the JPEG 2000 compression scheme is not designed for lossless compression.

Lossy mode

Our lossy compression scheme provides an efficient way to compress an HDR image in a low bit rate and still keep a high compressed image quality. In Table 2 we compare the results of our compression at various com-

Table 1. Storage requirements of different HDR image formats.					
Lossless Schemes	**Relative Error ($\times 10^{-3}$)**	**Image 1 (Mbytes)**	**Image 2 (Mbytes)**	**Image 3 (Mbytes)**	**Image 4 (Mbytes)**
Our lossless scheme	See notes*	3.3	1.6	1.3	11.6
RGBE (RLE)	10	2.7	1.3	1.1	10.9
LogLuv (32 bits)	3	2.5	1.2	0.7	7.1
OpenEXR (PIZ)	1	2.1	1.3	0.8	7.2

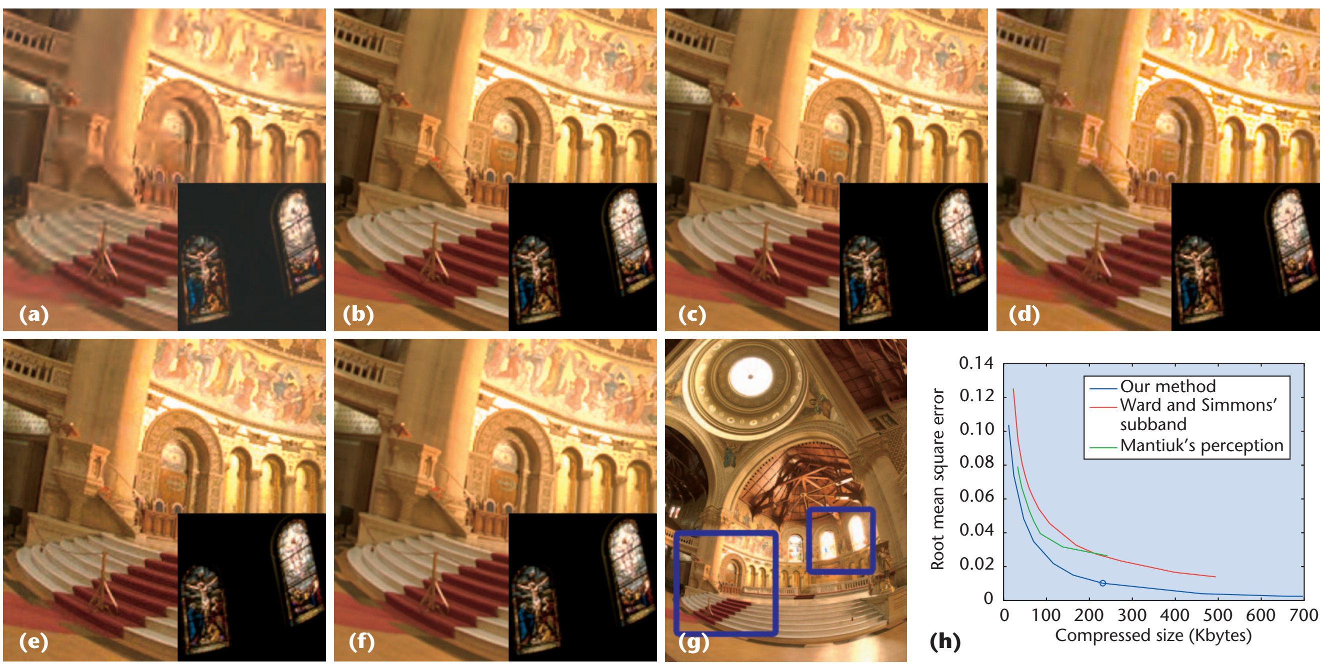
*The relative error of our method depends on the actual dynamic range. For the four images used in this table whose dynamic ranges are 4.2, 5.9, 3.6, and 4.8, respectively, the relative errors are 0.15×10^{-3}, 0.21×10^{-3}, 0.13×10^{-3}, and 0.17×10^{-3}, respectively.

Table 2. Compression statistics and comparison with Ward and Simmons' and Mantiuk's methods.

Method	**Size (Kbytes)**	**Root Mean Square Error**	**Visual Difference Predictor ($\times 10^{-3}$)**	**Compression/ Decompression Times (Seconds)**
Our method (rate = 0.01)	23	0.074	96	1.7/0.64
Our method (rate = 0.05)	118	0.022	34	1.7/0.68
Our method (rate = 0.10)	230	0.010	23	1.7/0.70
Ward and Simmons'	125	0.040	46	1.1/0.58
Mantiuk's	138	0.032	44	N/A

pression ratios with the compression result using Ward and Simmons' and Mantiuk's compression schemes. We obtained the experimental data for Mantiuk's method from the compressed images provided by its author. We acquired the data for Ward and Simmons' method from the implementation of their method (provided by its authors), and we ran it using the parameters ("–a 0.67 –b 0.75 –c full") provided in a demo from the authors.

The data in Table 2 are the results from applying the various methods to the image in Figure 3. This 512 × 768 pixel HDR image occupies 1.1 Mbytes in RGBE (RLE, or run length encoding) and 823 Kbytes in OpenEXR (PIZ, a wavelet compression) format (for more information, see http://www.openexr.com/). The timing quoted in this

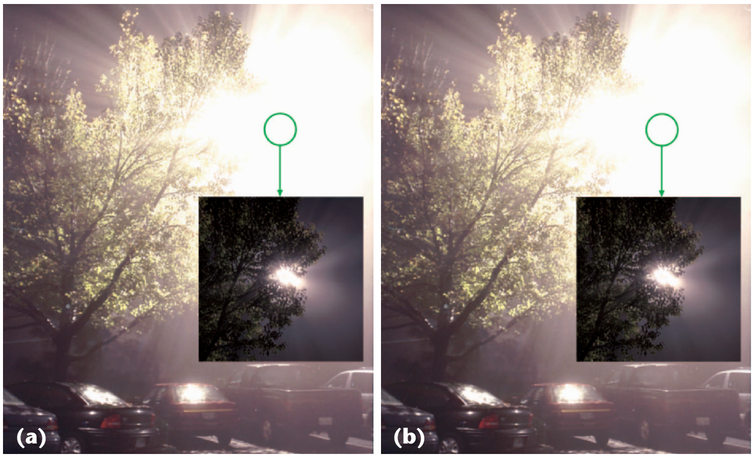


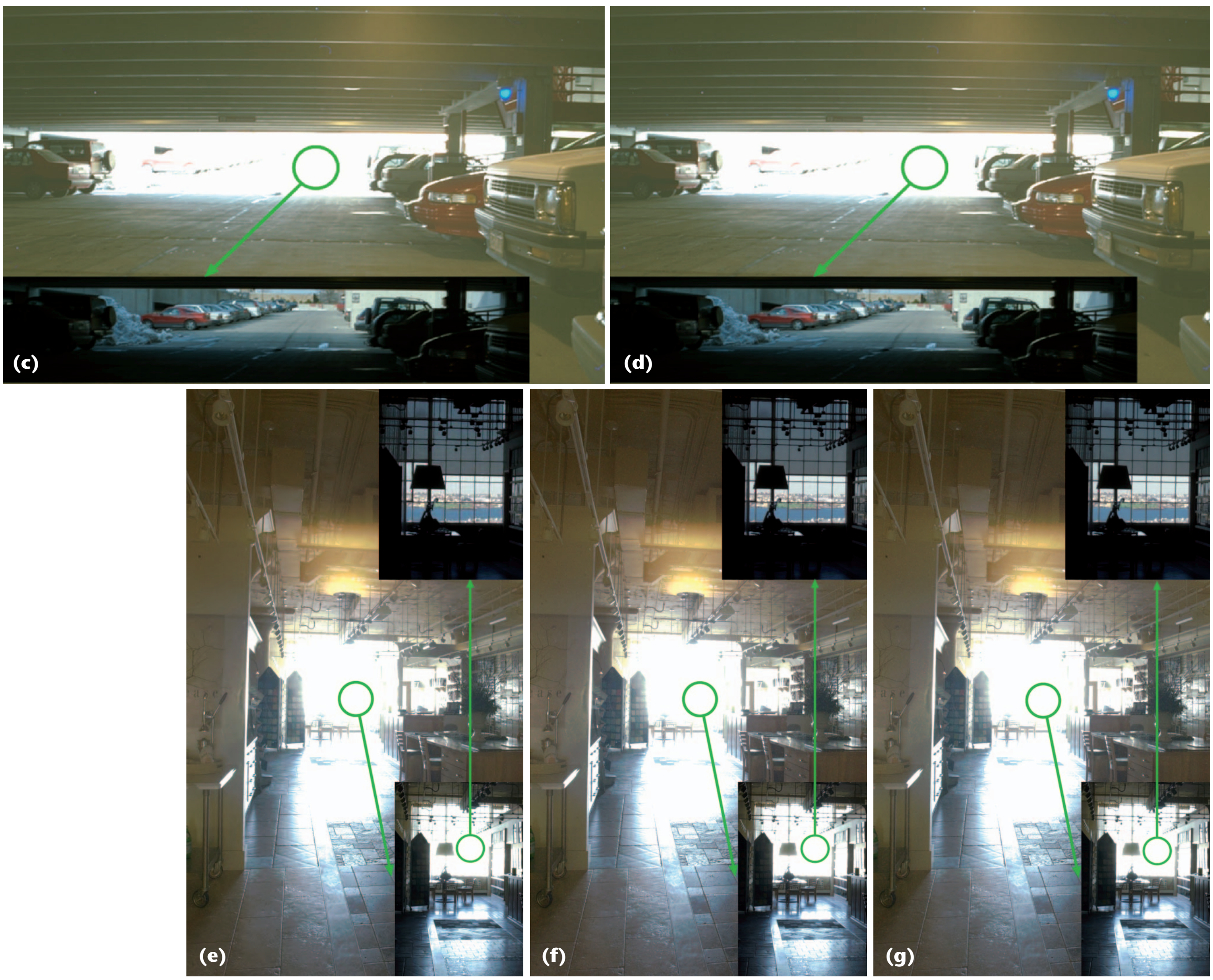
3 **Visual quality comparison: The rate for (a)–(c) is 0.01, 0.05, 0.10, respectively is 23, 118, and 230 Kbytes. (d) Compressed image using Ward and Simmons' subband encoding (125 Kbytes). (e) Compressed image using Mantiuk's perception-based encoding (138 Kbytes). (f) Reference image. (g) Relative positions of background images and the insets in (a)–(f) shown as blue boxes. (h) Comparison of HDR images using lossy compression.**

table is from the runtime on an Intel Xeon 1.7G PC with 1 Gbyte of memory and running Windows XP.

Figure 3 shows the results of the compression. Rate is a parameter in JPEG 2000, which specifies the ratio of desired data size to raw data size. The raw data size is the number of pixels times the number of bytes per pixel (for example, 6 for $n = 16$). Figure 3h shows the square root of the mean square error between the compressed image and the reference image in the logarithm domain. To have a metric that correlates with subjective perception, we use Lubin's visual difference predictor (VDP)[9] and the mean value of the difference map as a visual fidelity indicator in this article. In the last column of Table 2, we show the compression and decompression times in seconds. You can see that our algorithm consumes a considerable amount of time, but this is not a problem in encoding a single static HDR image.

The compressed image in Figure 3a shows some blur artifacts, but the compressed images in Figures 3b and 3c are indistinguishable from the reference image in Figure 3f. In comparison, Figure 3d shows the compression result of Ward and Simmons' subband encoding scheme,[2] and Figure 3e uses Mantiuk's perception-based encoding.[3] In this image we see visible artifacts in the brighter areas of the scene. The visual differences agree with the error-predicted VDP. Thus, keeping the





4 Lossy compressed results. (a) Compressed image (248 Kbytes; rate = 0.05) of (b) reference image (2,683 Kbytes in RGBE RLE). (c) Compressed image (108 Kbytes; rate = 0.05) of (d) reference image (1,142 Kbytes in RGBE RLE). (e) Compressed image (154 Kbytes; rate = 0.01) of (f) reference image (766 Kbytes, rate = 0.05) and (g) reference image in RGBE RLE (8,356 Kbytes).

visual quality the same, our scheme produces compressed images at a bit rate of about one-fifth of that achieved using Ward and Simmons' subband encoding.[2]

Table 1 shows the performance of our lossless compression scheme with that of the other available methods. Although compressed images obtained using our method have the smallest relative error, they are the bulkiest of the lot. OpenEXR is a clear winner in terms of compressed image size. However, we must note that the quoted average compression ratio—which is the average ratio between raw image size and compressed image size—for OpenEXR is only 2:1 (see http://www.openexr.com/) and hence is not at all suitable in applications requiring aggressive compression performance. So there is a clear need for a lossy compression scheme. We show in the following that our lossy scheme excels by providing the best compression ratio with the least amount of relative error.

In Table 2, we also compare our lossy compression with Ward and Simmons' subband encoding and Mantiuk's perception encoding, investigating the compression quality in terms of the root mean square error changing with compressed size, as Figure 3h shows. Our method has obvious advantages, especially in small compressed sizes (a low bit rate).

Figure 4 shows the compression results for three other HDR images. For the image in Figure 4c, our HDR image encoder compresses the original HDR image (1,116 Kbytes in RGBE format) to 108 Kbytes without introducing any visually distinguishable differences. In fact, in all the test HDR images in this figure (Figures 4a, 4c, and 4f), compression rates greater than or equal to 0.05 produce results that are visually indistinguishable from the original HDR images (Figures 4b, 4d, and 4g).

Figure 4e shows that our lossy compression performs quite well for very low bit rate and very high dynamic range. Visit http://graphics.cs.ucf.edu/hdri/#j2 to view additional results.

Conclusions

Our method extends JPEG 2000 to compress HDR images. It thus acquires other benefits from JPEG 2000, like scalability, error resilience, and region of interest. Our wavelet-based approach is superior to any discrete cosine transform-based one; for example, it does not exhibit blocking artifacts. In contrast, this artifact issue is a serious problem for Mantiuk's and Ward and Simmons' methods under low bit rates.

Compared to other lossy HDR compression schemes, our approach can reach the same visual quality at a much lower bit rate. It enables a minimum coding error in the logarithm domain with any bit rate budget. For the HDR image in Figure 3, even a 23-Kbyte image is enough to achieve a visually good result.

The quantization error that our approach introduces is limited by the actual dynamic range and the maximum bit depth of the JPEG 2000 implementation. The highest precision in the log domain using JasPer will be $R/(2^{16+1}-2) = R*0.00076$ percent, where R is the actual dynamic range. For most natural HDR images, whose dynamic range covers up to nine orders of magnitude, the pixel coding error in the logarithm domain is no more than 0.007 percent, with a corresponding relative error, according to Equation 5, of 0.03 percent. This is much less than 0.1 percent, the precision (relative error) of the *half* data type (a new 16-bit floating data type first introduced in Nvidia's GPU) used in OpenEXR. The reason our pixel encoding can have a higher precision than OpenEXR, while using the same number of bits, lies in the fact that we use the actual dynamic range, rather than the nominal dynamic range of the half type.

The lossless mode has a larger bit rate than OpenEXR, and even more than JPEG-LS on average.[6] But the lossy mode is superior to others, particularly in low bit rates. Our approach provides a simple, straightforward, and efficient lossy HDR encoding.

Future directions

Eventually, we would like the color transformation to occur before doing the log transformation. However, our attempt in doing so has brought about color artifacts in the dynamic range areas. Currently we aren't certain about the reason, but we believe that the issue could be in using the sRGB to YC_bC_r transformation, which is designed mostly for LDR images. As a part of our future work, we would like to find out the best uncorrelated color space and the appropriate transformation matrix.

One simple improvement we can make involves optimizing the logarithmic operation by taking advantage of the format definition of floating-point numbers whose codes include a mantissa and an exponent.

The choice of using a view-independent quantization is deliberate in our HDR image encoding scheme. The reason is that human eyes are not at a fixed adaptation level when viewing HDR images and hence will warrant an adaptive quantization for the wavelet coefficients as a function of pixel position. Although it is not impossible to address this, such consideration requires careful research that can build on previous work concerning adaptive quantization of conventional images.[10,11] The human eye is more sensitive to luminance than chrominance, which is exploited by the JPEG standard by subsampling the chrominance channel. It's also possible to encompass this property in the HDR image encoding for further optimization. In future research, we will study incorporating the knowledge of visual perception into our compression scheme.

It's possible to extend our approach to compress HDR video based on MJPEG 2000 (see Part 3 of JPEG 2000[7]). We can compress HDR video by sending each single frame to our HDR image compressor. However, the decoding time of JPEG 2000 is rather slow (0.5 second for a 512×768-pixel HDR image). The GPU implementation of JPEG 2000 is much faster (see http://www.cse.cuhk.edu.hk/~ttwong/demo/dwtgpu/dwtgpu.html), and can be used to develop a real-time HDR video codec. ■

Acknowledgments

We performed the research presented here in participation with the Research in Augmented and Virtual Environments supported by the Naval Research Laboratory VR Lab. It's also supported by the US Army's Science and Technology Objective Embedded Training for

Dismounted Soldiers (STO-ETDS) at the Research, Development and Engineering Command (RDECOM) in Orlando, Florida, and by ATI Research.

Figures 4a and 4b are courtesy of Jack Tumblin, Northwestern University. Figures 4e through 4g are courtesy of Byong Mok Oh.

References

1. E. Reinhard et al., *High Dynamic Range Imaging*, Morgan Kaufmann, 2005.
2. G. Ward and M. Simmons, "Subband Encoding of High Dynamic Range Imagery," *Proc. 1st Symp. Applied Perception in Graphics and Visualization* (APGV), ACM Press, 2004, pp. 83-90.
3. R. Mantiuk et al., "Perception-Motivated High-Dynamic-Range Video Encoding," *ACM Trans. Graphics*, vol. 23, no. 3, 2004, pp. 733-741.
4. G.W. Larson, "LogLuv Encoding for Full-Gamut, High-Dynamic Range Images," *J. Graphics Tools*, vol. 3, no. 1, 1998, pp. 15-31.
5. S.S. Stevens and J.C. Stevens, "Brightness Function: Parametric Effects of Adaptation and Contrast," *J. Optical Soc. of America*, vol. 50, no. 11, 1960, p. 1139A.
6. M. Rabbani and R. Joshi, "An Overview of the JPEG 2000 Still Image Compression Standard," *Signal Processing: Image Communication*, vol. 17, no. 3, 2002, pp. 3-48.
7. *Information Technology, JPEG 2000 Image Coding System—Part 1: Core Coding System*, ISO/IEC 15444-1:2000, Int'l Organization for Standardization/Int'l Electrotechnical Commission, 2000.
8. M.D. Adams and F. Kosentini, "JasPer: A Software-Based JPEG-2000 Codec Implementation," *Proc. IEEE Int'l Conf. Image Processing*, vol. 2, nos. 10-13, 2000, pp. 55-56.
9. J. Lubin, "A Visual Discrimination Model for Imaging System Design and Evaluation," *Visual Models for Target Detection and Recognition*, E. Peli, World Scientific Publishers, ed., 1995, pp. 245-283.
10. M.J. Nadenau and J. Reichel, "Compression of Color Images with Wavelets under Consideration of the HVS," *Proc. IS&T/SPIE Conf. Human Vision and Electronic Imaging IV*, vol. 3644, SPIE, 1999, pp. 129-140.
11. T. Strutz, "Adaptive Quantization for Lossy Image Compression Controlled by Noise Detection," *Proc. Data Compression* (DCC), IEEE CS Press, 2001, p. 517.

Ruifeng Xu *is a PhD student in the School of Computer Science, University of Central Florida. His research interests include global illumination, realistic rendering in real time, and HDR video compression. Xu received an MS in computer science from Zhejiang University, China. Contact him at rxu@cs.ucf.edu.*

Sumanta N. Pattanaik *is an associate professor of computer science at the University of Central Florida. His research interests include real-time realistic rendering. Pattanaik has a PhD in computer science from Birla Institute of Technology and Science, Pilani, India. He is a computer graphics editor of* ACM Computing Reviews *and is a member of the IEEE. Contact him at sumant@cs.ucf.edu.*

Charles E. Hughes *is a computer science professor at the University of Central Florida where he is also a member of the CS Graphics Group and the Media Convergence Laboratory. His research interests include mixed reality, distributed interactive simulation, and models of computation. Hughes has a BA in mathematics from Northeastern University, and an MS and a PhD in computer science from Pennsylvania State University. Contact him at ceh@cs.ucf.edu.*

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