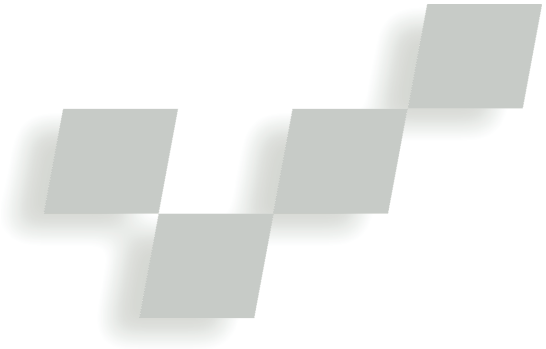


Interactive Relighting of Panoramas

Tien-Tsin Wong and Pheng-Ann Heng
Chinese University of Hong Kong

Chi-Wing Fu
Indiana University at Bloomington

The main advantage of image-based computer graphics is that the rendering time is independent of scene complexity, because rendering is a process of manipulating image pixels instead of simulating light transport. The major drawback is its rigidity. Once we capture a scene as images, we can no longer modify them. The ability to control a modeled scene's illumination would certainly enhance the 3D illusion, which in turn improves viewers' understanding of the environment.

Our panoramic image representation lets users incorporate the illumination information into the panorama and interactively adjust lighting conditions.

Due to its simplicity, efficiency, and visual richness, panoramic image representation (which uses a single panoramic image to provide an immersive illusion within an environment) has been successfully adopted by industry in various commercial applications. Unfortunately, it inherits the same rigidity. However, if we can modify the illumination by *relighting* the images instead of rendering the geometric models, the time for image synthesis should also be independent of scene complexity. (By relighting, we mean generating desired images with a novel lighting condition from prerecorded reference images.) This will save artists and designers enormous time in fine tuning the lighting condition to achieve dramatic and emotional atmospheres.

Researchers have proposed a few relighting techniques in the framework of image-based modeling and rendering. Nimeroff et al.[1] described an efficient technique to relight images under various natural illumination. Their intrinsic assumption of natural illumination might not be applicable for other kinds of illumination. Another approach for capturing illumination information[2] is using singular value decomposition (SVD)[3] to extract the principle components (eigen-images) from reference images. Unfortunately, because no direct relationship exists between the lighting geometry (the light vector's direction) and the principle components extracted, the illumination is uncontrollable. That is, we can change the illumination but can't precisely specify the lighting direction.

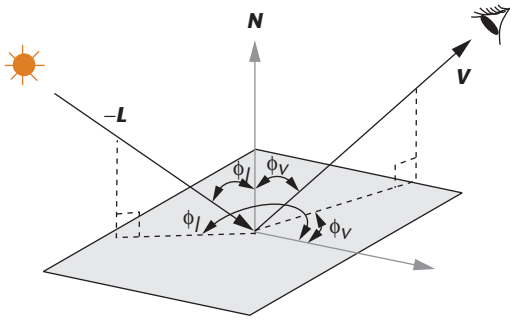
All these reasons motivated us to develop an interactive image-based panorama viewer that incorporates illumination information for relighting. We propose a concept of an apparent bidirectional reflectance distribution function of a pixel (pBRDF)[4] to represent the outgoing radiance distribution passing through the pixel window on the image plane. By treating each image pixel as an ordinary surface element, we can record its radiance distribution under various illumination conditions in a table. If we incorporate this table into the panoramic image data structure, we can relight the image. We developed an interactive panorama viewer that allows panning, tilting, and zooming as well as modifying the modeled environment's lighting condition.

Apparent BRDF of a pixel

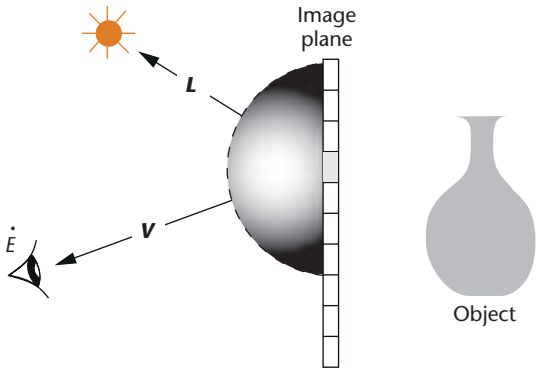
We can calculate the reflected radiance leaving the surface element once we know the viewing vector, light vector, incident radiance, and surface element's reflectance. The most general form for expressing a surface element's reflectance is the bidirectional reflectance distribution function (BRDF), which is a 4D table indexed by the light vector $\boldsymbol{L}$ and the viewing vector $\boldsymbol{V}$ (see Figure 1). It tells us how the surface element looks from the viewing direction $\boldsymbol{V}$ and when a light ray coming along the vector $-\boldsymbol{L}$ illuminates it. (We use the negative sign to be consistent with the convention that all vectors are denoted to be originated from the surface element.)

However, unlike in geometry-based computer graphics, we can't access the geometry details or the scene objects' surface properties. Accessing the geometric details implies that the rendering time complexity becomes dependent on the scene complexity. Nevertheless, we can adopt the same analogy as BRDF to express how a pixel (instead of a surface element) will look from $\boldsymbol{V}$ when a light ray coming along $-\boldsymbol{L}$ illuminates the scene behind the pixel window. We can regard this as *image-based reflectance*, or pBRDF. It is actually the aggregate reflectance of all surface elements that are visible through the pixel window.

0272-1716/01/$10.00 © 2001 IEEE



1 Geometric meaning of the notations of parameters in a bidirectional reflectance distribution function.



2 Treating each pixel on the image plane as an ordinary surface element and measuring its apparent BRDF.

Therefore, we treat each pixel on the image plane as an ordinary surface element and measure and record its apparent BRDF. Figure 2 illustrates the idea in a 2D cross section. The vectors $\boldsymbol{L}$ and $\boldsymbol{V}$ are the light and viewing vectors, respectively. Position $\dot{E}$ is the center of the projection (or eye). The apparent BRDF is a table indexed by the vectors $\boldsymbol{V}$ and $\boldsymbol{L}$, or the quadruple $(\theta_v, \phi_v, \theta_l, \phi_l)$.

The standard definition of a real surface element's BRDF is

$$\rho(\theta_v,\phi_v,\theta_l,\phi_l) = \frac{L_r(\dot{p},\theta_v,\phi_v)}{L_r(\dot{p},\theta_l,\phi_l)\cos\theta_l d\omega}$$

where ρ is the surface element's BRDF, (θ_v, ϕ_v) specifies the viewing direction $\boldsymbol{V}$ in a spherical coordinate, (θ_l, ϕ_l) is the light vector $\boldsymbol{L}$ in a spherical coordinate, $\dot{p}$ is the position of element, $L_r(\dot{p},\theta,\phi)$ is the radiance along the vector passing through p in the direction (θ, ϕ)—we use subscript r to distinguish this from the vector $\boldsymbol{L}$, and $d\omega$ is the differential solid angle.

BRDF is the ratio between the radiance along the viewing direction and the radiance along the light vector weighted by the projected solid angle (the term $\cos\theta_l d\omega$). The projected solid angle is the area of differential solid angle after projecting onto the plane that contains the surface element. Because the differential solid angles might not be equal, the projected differential solid angle is mainly used to account for such unequal weight. In our first attempt[4] to include illumination information for the two-plane parameterized image-based representation,[5,6] we adopted this definition. (Yu and Malik[7] also adopted it to capture the reflectance of architectural models.)

However, defining the pBRDF this way might not be useful. Since the pixel isn't a true surface element but an imaginary window in the space, it doesn't physically reflect light energy. The actual reflection takes place at the real surfaces behind the pixel window. It's meaningless to scale the incident radiance by the projected differential sold angle, which is projected onto the image plane. Hence, we define the pBRDF as the ratio between the radiance along the viewing direction and the radiance along the light vector:

$$\rho_{\text{pixel}}(\theta_v,\phi_v,\theta_l,\phi_l) = \frac{L_r(\dot{p},\theta_v,\phi_v)}{L_r(\dot{p},\theta_l,\phi_l)}$$

This new definition simplifies the computation for relighting images, because it drops the cosine term.

In the panorama application, the viewpoint is fixed in the space. Although the viewing vectors of different pixels in the panoramic image differ, they keep constant whenever the viewpoint is fixed. Hence, we can drop the index (θ_v, ϕ_v) in the previous formulation:

$$\rho_{\text{pixel}}(\theta_l,\phi_l) = \frac{L_r^{\dot{E}}(\dot{p})}{L_r(\dot{p},\theta_l,\phi_l)} \quad (1)$$

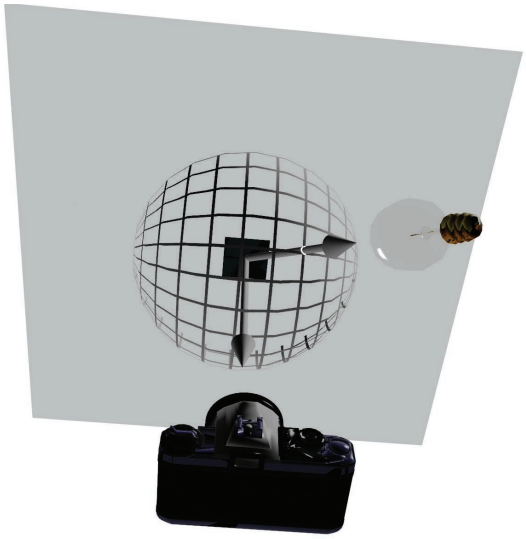
where $L_r^{\dot{E}}(\dot{p})$ is the radiance along the ray passing through the pixel window at $\dot{p}$, arriving our eye at $\dot{E}$. In other words, the pBRDF in the panoramic image is a spherical function that only depends on a light vector's direction.

Recording illumination information

To prepare the illumination information for a panorama, we need to fill the 2D spherical function with samples from the reference images. We must decide what kind of light source to use to illuminate the scene and how many images we need to take. The simplest way is to illuminate with a single directional light source, because its light vector stays constant for any point in the space. Note that the physical reflection isn't taking place at the pixel's location but at the surface elements behind the pixel window. Using a directional light source is more meaningful because the captured pixel value tells us what the surface elements behind the pixel window will look like when all surface elements are illuminated by parallel light rays in the direction of $\boldsymbol{V}$.

Another issue is how to take sample panoramic images. It is natural and simple to take a sample with the light vector oriented on the spherical grid as Figure 3 (next page) shows.

When measuring the pBRDF, it's convenient to specify the direction in a local coordinate system. Figure 4a (next page) shows that each pixel in a planar image is associated with a local coordinate system. We must transform vectors to the local coordinate system during sampling and relighting. Moreover, the north poles of local frames are usually aligned with the image plane's normal. All coordinate systems are in the same orientation in the case of a planar image. The major advantage of using this orientation-aligned scheme is that a direc-



3 Capturing the illumination characteristics by orienting the light vector on the spherical grid.

tional light source's light vector will always be transformed to the same local vector for any pixel. Hence, we only need one transformation.

In the case of a panoramic image, projection manifold isn't planar but spherical or cylindrical (see Figure 4b). If each pixel's local coordinate system still aligns with the normal of a projection manifold, the local coordinate systems will not be in the same orientation. In other words, we must do different transformations of the light vector for different pixels, which is obviously inefficient. There's no need to align the local coordinate system with the normal of a projection manifold. Instead, we can align all local coordinate systems to a global frame as Figure 4c shows. Unlike the BRDF, which describes reflectance over the hemisphere, we must record the reflectance over the whole sphere since the real surface elements behind might not be parallel to the imaginary pixel window.

Therefore, for each lighting direction, or grid point (θ_l, ϕ_l) on the spherical coordinate system, we generate a panoramic image $I_{\theta_l\phi_l}$. The sampling rate on the spherical grid depends on the rate of radiance change on the illuminated scene, which in turn depends on the surface properties and scene complexity. An empirical sampling rate is 70×140. We use the value $v_{\theta_l\phi_l}(x, y)$ of pixel (x, y) in the image $I_{\theta_l\phi_l}$ to calculate the pBRDF when the light ray coming along (θ_l, ϕ_l) illuminates the whole scene by using the following equation (which we derive from Equation 1):

$$\rho_{\text{pixel}}(\theta_l, \phi_l) = \frac{v_{\theta_l,\phi_l}(x, y)}{L_r(\dot{p}, \theta_l, \phi_l)} \quad (2)$$

where v is the pixel value and $L_r(\dot{p}, \theta_l, \phi_l)$ is the radiance emitted by the directional light source in the direction (θ_l, ϕ_l).

Equation 2 assumes the pixel value is linear to the radiance, which might not be a problem for synthetic images. For real-world images, we can recover a quantity that is linear to the radiance from reference images using a technique Debevec and Malik proposed.[8] We can then use this quantity to replace the pixel value v in Equation 2.

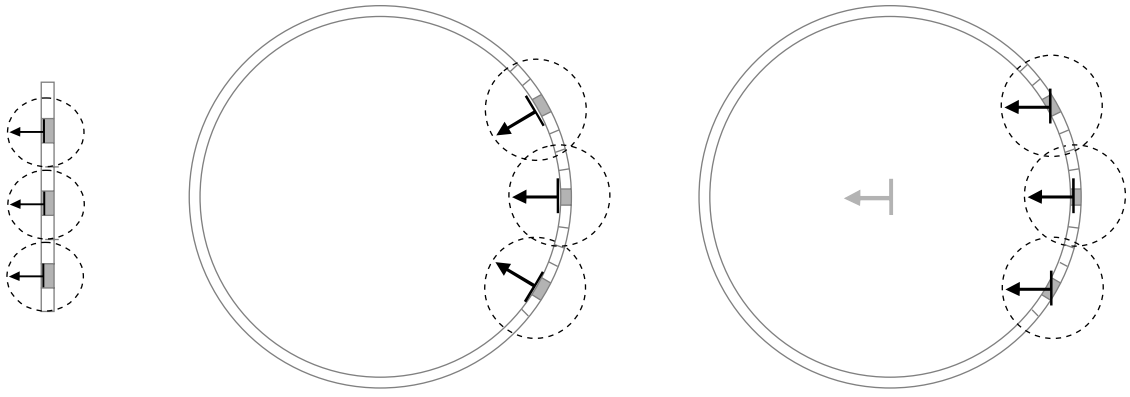
Relighting

Once we sample and store the pBRDFs, we can use them for relighting. Using the property of superposition,[9] we can compute each pixel's final radiance (or value). We propose a local illumination model for image-based rendering (Equation 3) that uses superposition to relight the image-based scenery under different lighting conditions.

$$\text{radiance through } p = \sum_i^n \rho_{pixel}(\theta_l^i, \phi_l^i) L_r(\dot{p}, \theta_l^i, \phi_l^i) \quad (3)$$

where n is the total number of light sources, (θ_l^i, ϕ_l^i) specifies the direction $\boldsymbol{L}_i$ of the ith light source, $\rho_{\text{pixel}}(\theta_l^i, \phi_l^i)$ is the reflectance of pixel (x, y) when the light vector is (θ_l^i, ϕ_l^i), $L_r(\dot{p}, \theta_l^i, \phi_l^i)$ is the radiance along (θ_l^i, ϕ_l^i) due to the ith light source, and $\dot{p}$ is the position of pixel (x, y).

Note that the illumination model is local. That is, it accounts for the direct radiance contribution from the light sources but not the indirect radiance contribution. This illumination model isn't the result of borrowing the



4 Alignment of local coordinate systems. (a) The local coordinate systems, for a planar image, are in the same orientation. For panoramas, we can align the local coordinate system to (b) the local normals or (c) a global frame.

existing illumination model but results from using the image superposition properties.

The pBRDF is actually an aggregate BRDF of all visible objects behind the pixel. Consider k unoccluded objects, which are visible through the pixel window and illuminated by n light sources. (If objects that occlude each other exist, we can always subdivide them into visible [unoccluded] and invisible [occluded] portions. Invisible portions will never contribute any radiance to the final image. Hence, we consider only those unoccluded objects without loss of generality.) The radiance along $\boldsymbol{V}$ is

$$\begin{aligned}&\sum_i^n \rho_i^1 L_r^i + \sum_i^n \rho_i^2 L_r^i + \ldots + \sum_i^n \rho_i^k L_r^i \\ &= \sum_j^k \rho_1^j L_r^1 + \sum_j^k \rho_2^j L_r^2 + \ldots + \sum_j^k \rho_n^j L_r^n \\ &= \rho_1 L_r^1 + \rho_2 L_r^2 + \ldots + \rho_n L_r^n \end{aligned} \quad (4)$$

where ρ_i^j is the reflectance of the jth object surface when illuminated by the ith light source; L_r^i is the short hand of $L_r(\dot{p}, \theta_i^j, \phi_i^j)$, the radiance due to the ith light source; and

$$\rho_i = \sum_{j=1}^k \rho_i^j$$

is the aggregate reflectance of k objects. We have recorded pBRDF.

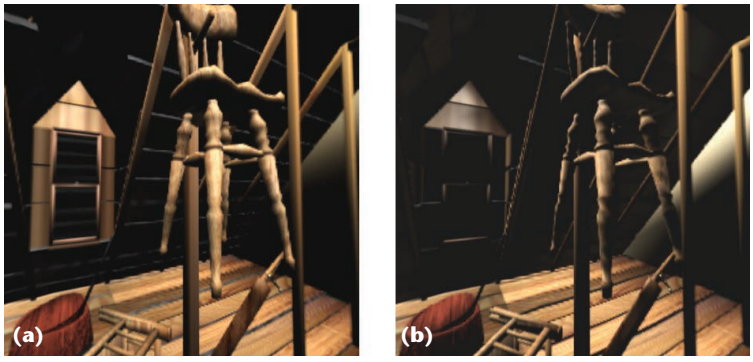
The first row in Equation 4 shows the sum of reflected radiances from all k unoccluded objects. Due to the linearity of illumination, we can reorder the terms to derive the result on the third row, which is the sum of multiplications of aggregate reflectance and the radiance emitted by each light source.

Lighting direction

With Equation 3, we can modify the lighting direction by substituting a different value for (θ_l, ϕ_l). Figures 5a and 5b show the perspective snapshots of a panorama with a directional light in two different directions. We don't require geometric model during the relighting.

Light intensity

We can also manipulate the light source's intensity in Equation 3 by assigning the value of L_r^i for the ith light source. Figure 6a shows another perspective snapshot of the same panorama illuminated by a blue spotlight.



5 Change of lighting direction. (a) Directional light cast on top of the hanging chair. (b) We modified the illumination to cast the same image-based scene with light from a different direction.

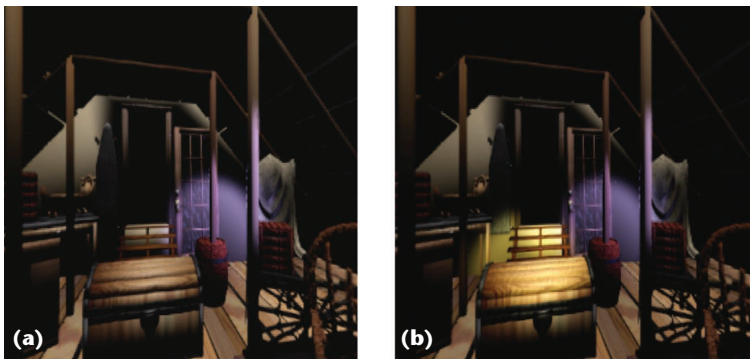
Multiple light sources

Because of the linearity of light transport, we can include an arbitrary number of light sources for the illumination. The trade-off is computational time. For each extra light source, we must compute an additional multiplication and addition in Equation 3. Figure 6b shows the same panorama illuminated by blue and yellow spotlights. Although we captured the reference images under the illumination of a single directional light source, we can illuminate the relighted images with multiple light sources, each with a different color.

Types of light sources

The desired light source for relighting is directional, because the reference images are captured under the illumination of a directional illuminator. Relighting with a directional light source is efficient for evaluating Equation 3 because the same light vector θ_l^i, ϕ_l^i illuminates all pixels in the same image. We only need one transformation to transform the light vector to a pixel's local frame. Moreover, we don't require geometry information for relighting, because we capture all reference images under a directional light source.

However, relighting isn't restricted to directional light. We can extend to a point source, spotlight, or more general area light source. It will be more expensive to evaluate Equation 3 for other types of light sources, because we must determine the light vector θ_l^i, ϕ_l^i from pixel to pixel. Since the image plane where the pixels are located is only a window in a 3D space, the surface element that reflects the light may be located on any point along the ray $\boldsymbol{V}$, as Figure 7 (next page) shows. To determine the light vector $\boldsymbol{L}$ correctly for other types of light sources, we must first locate the



6 Multiple light sources. (a) A blue light illuminating the attic scene. (b) We added one more yellow spotlight.

intersection point of the ray and the object. With directional light sources, the light vectors are identical for any point in 3D space, which avoids this problem. One way to determine $\boldsymbol{L}$ is to use the depth image. While we can easily do this for synthetic scenes, real-world scenes can be more difficult. Using a range scanner or computer vision techniques might provide a solution.

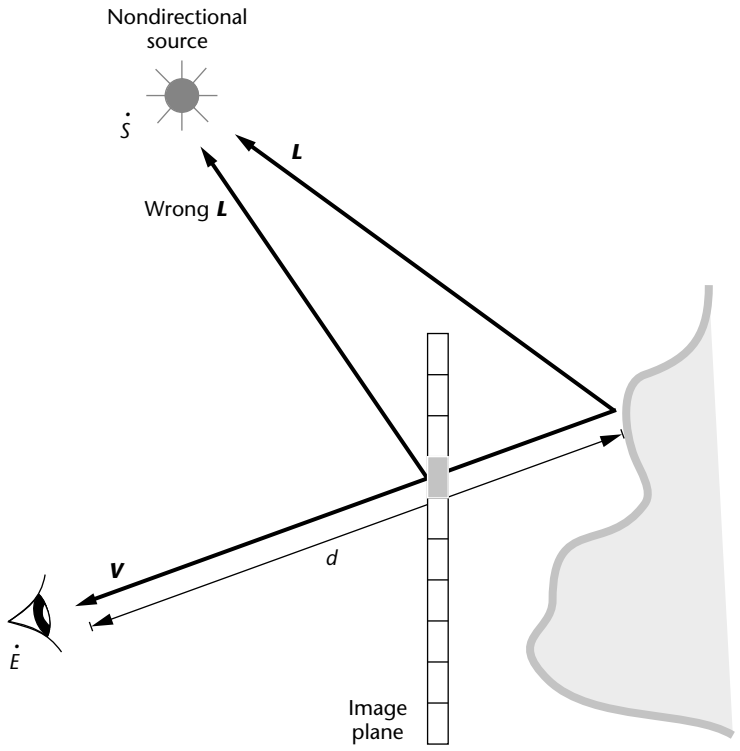
With the additional depth map, we can correctly relight the scene with any type of light source by finding the correct light vector $\boldsymbol{L}$ using the following equation:

$$\boldsymbol{L} = \dot{S} - \dot{E} + \frac{\boldsymbol{V}}{|\boldsymbol{V}|} d \qquad (4)$$

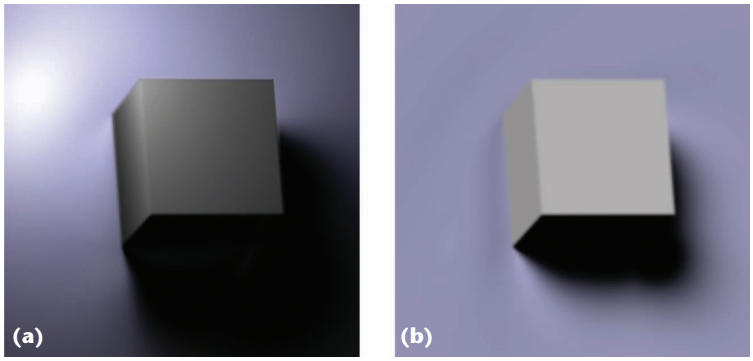
where $\boldsymbol{L}$ is the light vector, $\dot{S}$ is the nondirectional light source's position, $\dot{E}$ is the eye's position, $\boldsymbol{V}$ equals $\dot{E}-\dot{p}$ and is the viewing direction, and d is the depth value.

The illuminator can be a point source, spotlight, or slide projector source. Figure 8 shows an image-based scene containing a box lying on the plane. A point source (Figure 8a) and a directional source (Figure 8b) illuminate the scene. We recorded all input reference images with a single directional light source. Surprisingly, with this extra depth information, we can relight the image-based scene with other types of light sources. Even though we need a depth map, the relighting is still independent of the scene complexity, because the depth map is also an image with the same resolution as the panorama.

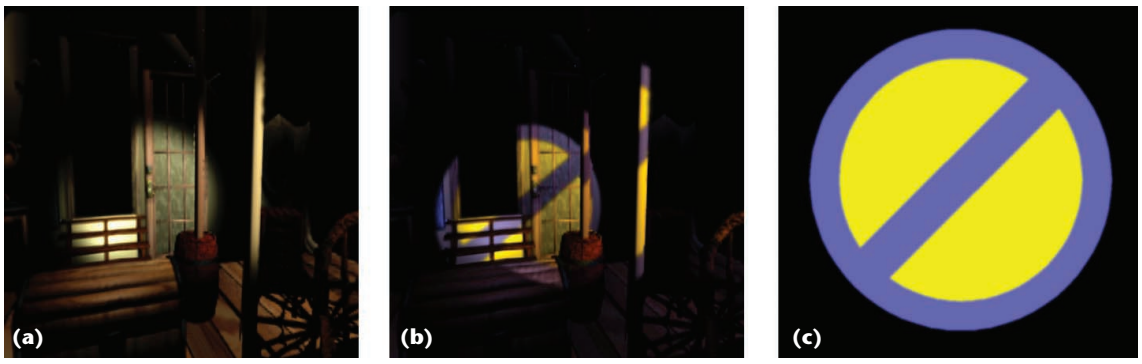
The relighted image can contain a shadow if we record it in the pBRDFs during sampling. Note the difference in the shadow cast by different sources in Figure 8. Figure 9a shows the result of relighting the attic scene with a spotlight. Note how the two pillars are correctly illuminated. Figure 9b shows the result of casting a slide (Figure 9c) onto the same scene. Theoretically,



7 Finding the correct light vector. To relight with a nondirectional illuminator, we need the depth value in the calculation of the correct light vector.



8 Point and directional light sources. (a) A shadow cast by a point source. (b) A shadow cast by a directional source.



9 Spotlight and slide projector sources. (a) A spotlight illuminates the scene. (b) The same scene illuminated by a slide projector source. (c) The slide we used for slide projection.

we can even illuminate the scene with an area light source by compromising the computational time.

Implementation and results

To demonstrate the feasibility and practicability of the proposed image representation, we implemented an interactive panorama viewer on both PC and Unix platforms. Figure 10 shows the viewer's user interface (SGI version). The lighting control panel on the left lets the user control various illumination parameters. The right window will then show the relighting result.

In our implementation, we use a cylindrical panorama instead of a spherical panorama to avoid the excessive sampling at the north and south poles. Since the proposed image representation is independent of a projection manifold, using a cylindrical panorama or spherical panorama makes no difference. To display the cylindrical panorama, we first map the relighted panorama onto the surface of a cylinder. The hardware graphics accelerator then draws the texture-mapped cylinder. Hence, we can pan and zoom in real time.

When the user modifies the lighting parameters, we recompute the pixel values using Equation 3. The software handles the entire computation.

At any given instance, only part of the panorama is visible through the display window, so we tend to postpone relighting the whole panorama—an approach we call *lazy relighting*. When the user changes the illumination, only the visible portion of the panorama is relighted, as the unfolded panorama in Figure 11 shows. The invisible portion will be relighted only when the user finishes the modification of lighting parameters.

Timing statistics

Table 1 shows the timing statistics of relighting on both PC and SGI platforms. We tested two complex scenes—an attic and city. The attic scene contains more than 500,000 triangles and requires more than 2 minutes to render using Alias|Wavefront on an SGI Octane with MIPS 10,000 CPU. Using the image-based approach, we can relight the scene interactively on the same machine, even though the software handles the relighting. The render column shows the rendering time of both scenes. The relight (PC) column shows the average relighting time (of the whole panorama) for the same scenes on a 667-MHz Pentium III. Similarly, the relight (SGI) column shows the average relighting time (of the whole panorama) on an SGI Octane. Since the hardware does the panning, tilting, and zooming, we didn't include their timing statistics in the table.

For a fair comparison, we did all the testing (both geometry-based rendering and image-based relighting) by illuminating the scenes with a single directional light source. Since both image-based scenes have a 1024×256 resolution, their relighting time is the same.

(a)

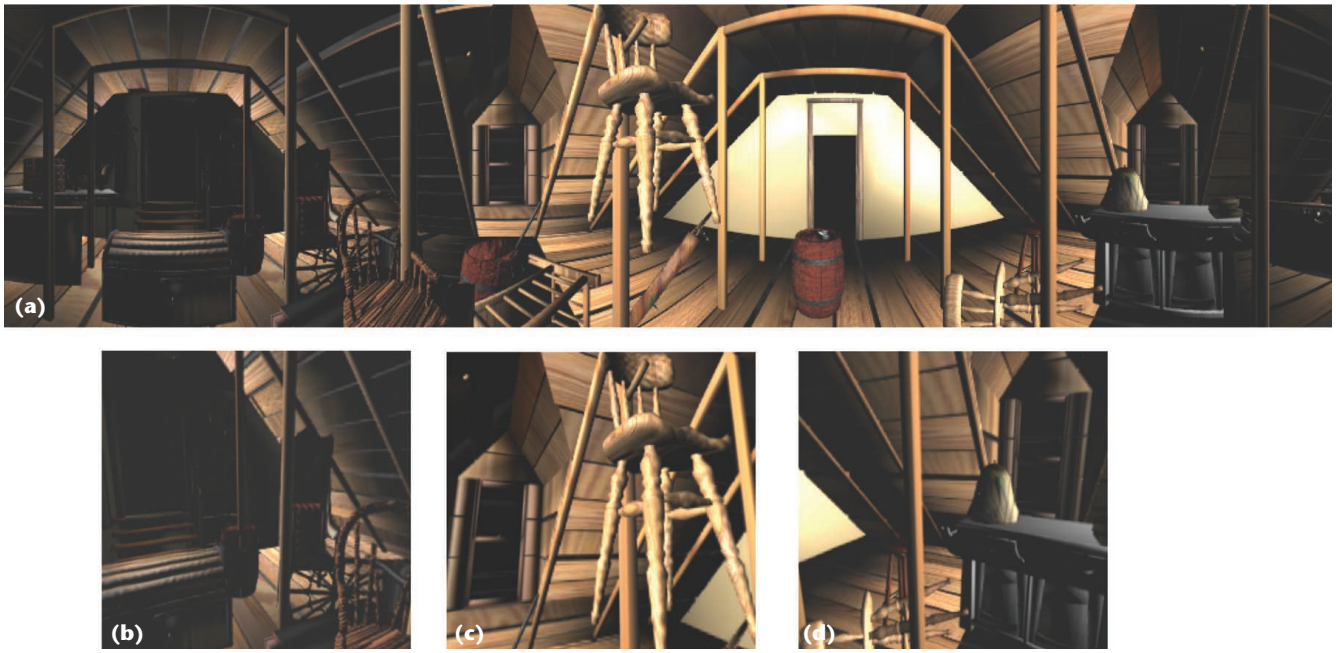
(b)

10 A panoramic viewer with controllable illumination. (a) Control panel and (b) the relighting result.

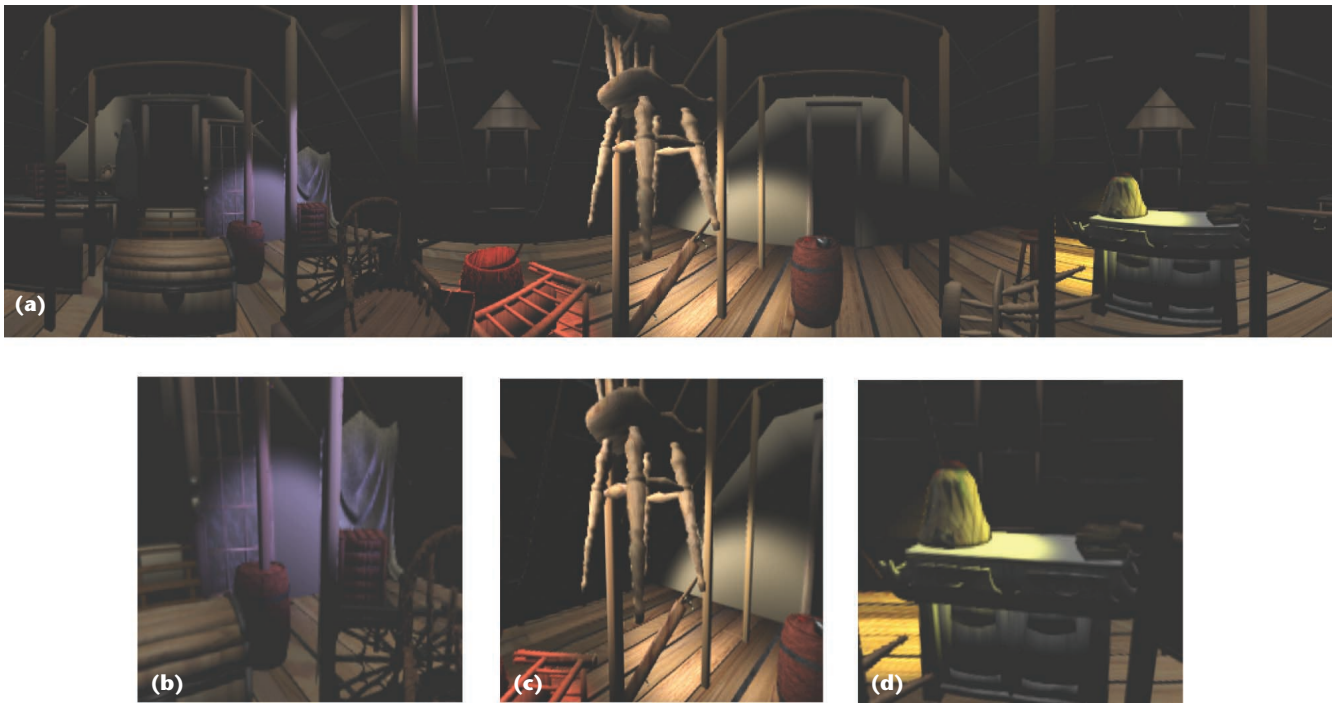
Table 1. Timing Statistics

Scene	Render (SGI)	Relight (PC)	Relight (SGI)
Attic	133 seconds	0.661 seconds	1.27 seconds
City	337 seconds	0.661 seconds	1.27 seconds

11 Lazy relighting. Only the visible portion (the region in the white box) relights during user modification.



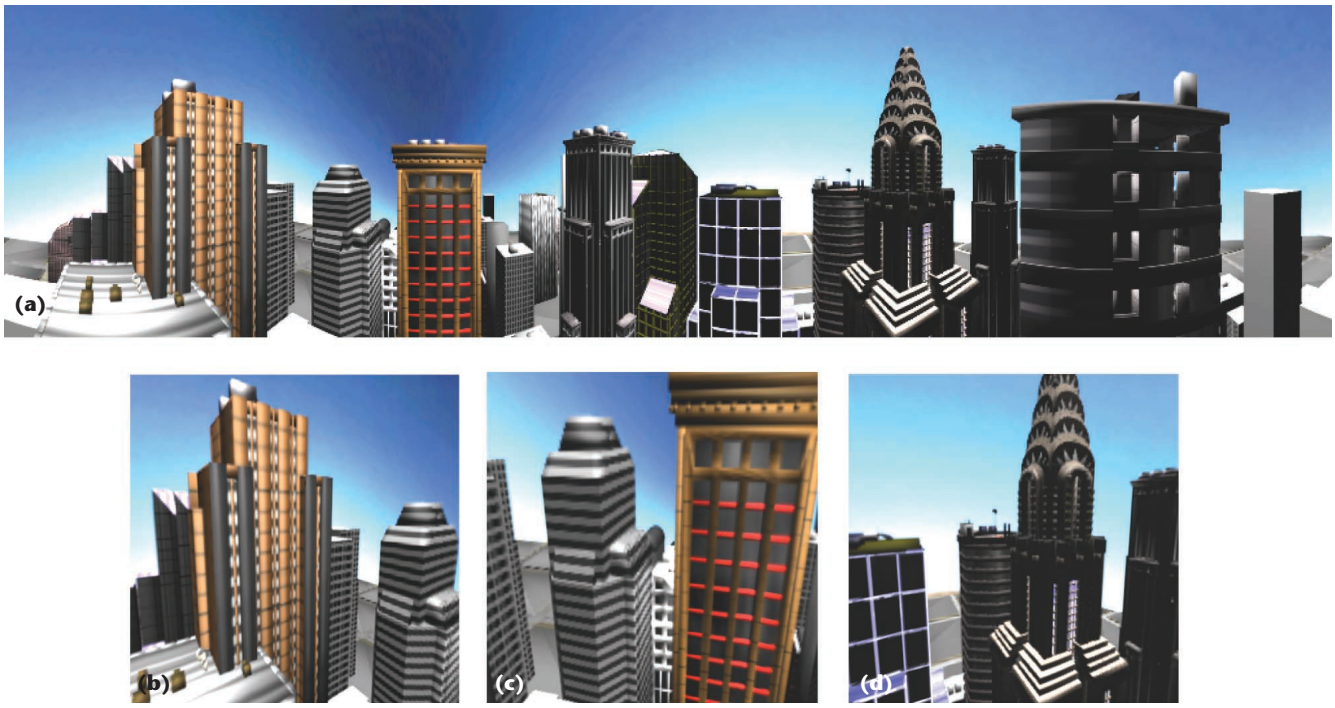
12 Attic. (a) Unfolded panorama. (b), (c), and (d) Perspective snapshots. (Relighting time: 0.661 second.)



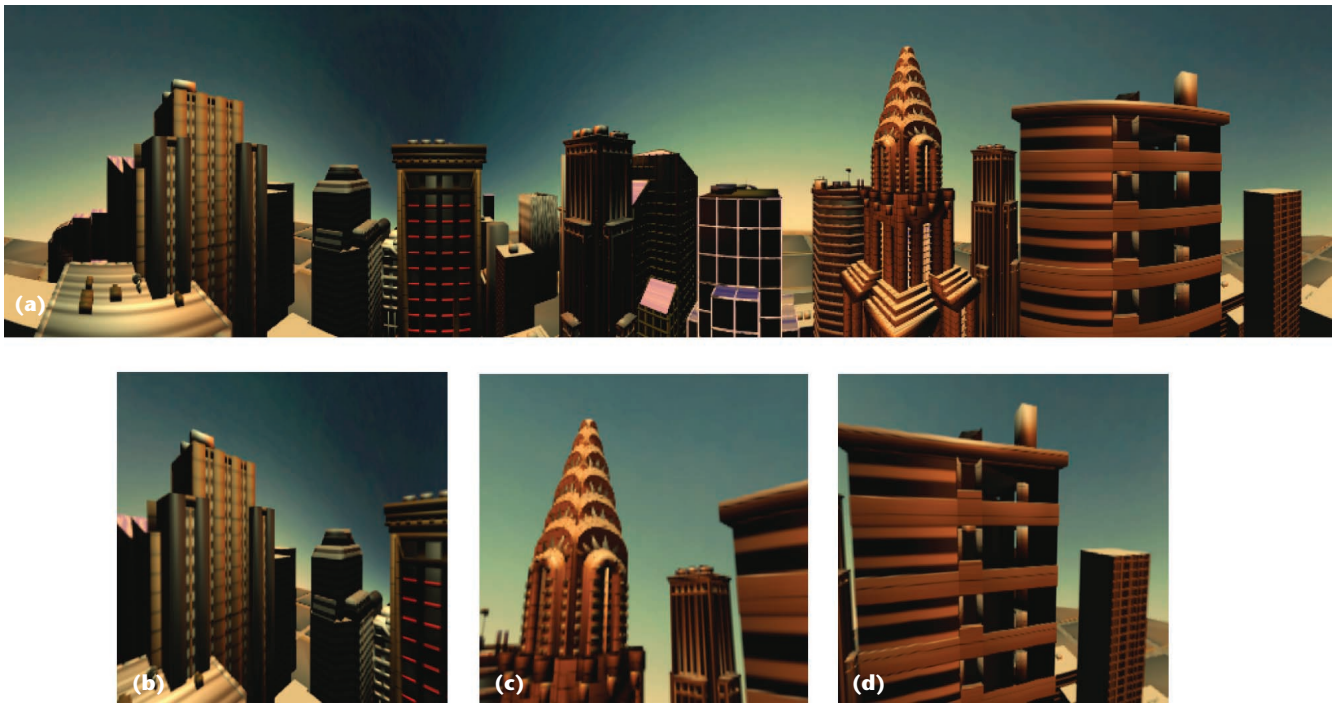
13 Attic illuminated by five spotlights. (a) Unfolded panorama. (b), (c), and (d) Perspective snapshots. (Relighting time: 10.305 seconds.)

Figures 12 through 15 show the relighting results of these two data sets. Figures 12a and 13a show the unfolded cylindrical panoramas of the attic scene relighted with a directional light source and spotlights, respectively. Figures 12b, 12c, and 12d and 13b, 13c, and 13d are the perspective snapshots of the panorama, which we obtained by warping the panorama. Note how the spotlights correctly illuminate the pillars and chair in the image-based attic.

Figures 14 and 15 show the results of relighting the image-based city scene. A directional light source illuminates both images. Figure 14 mimics the natural illumination of a clear skylight, while Figure 15 mimics the illumination during sunset. Each figure's caption gives the time it takes to relight the whole panorama (on a Pentium III). We don't need a depth map in the synthesis of Figures 14 and 15, because we only used directional light sources.



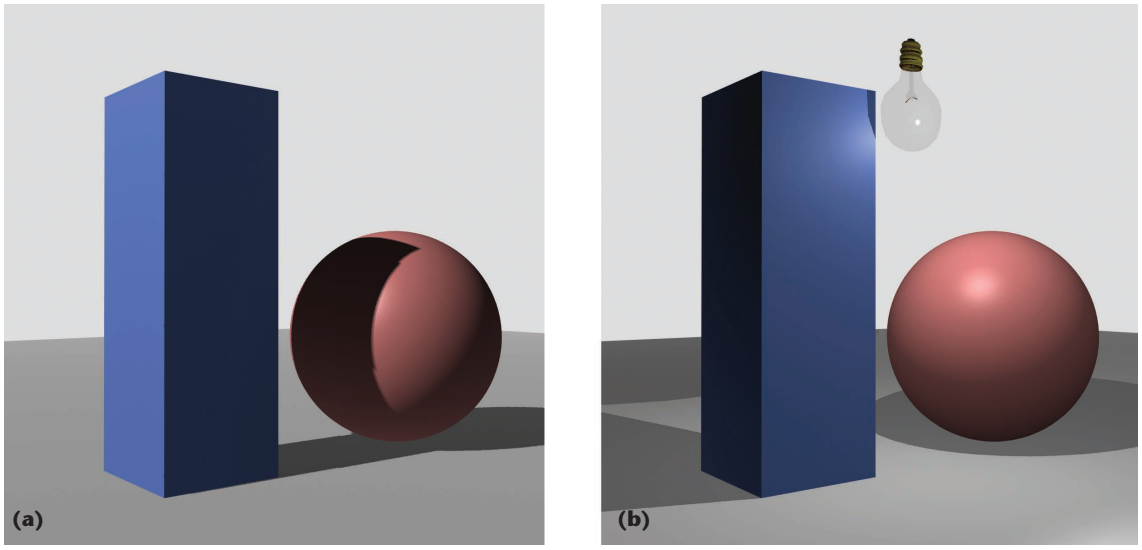
14 City scene under clear skylight. (a) Unfolded panorama. (b), (c), and (d) Perspective snapshots. (Relighting time: 0.661 second.)



15 Sunset city. (a) Unfolded panorama. (b), (c), and (d) Perspective snapshots. (Relighting time: 0.661 second.)

Compression

To reduce the data storage of image data, we compressed the pBRDF using spherical harmonics[10]—a common BRDF compression approach.[11] For each pixel, we transform its spherical reflectance table to a coefficient vector. The higher the vector's dimension, the more accurate the representation. Normally, 25 coefficients suffice to represent the reflectance table. With spherical harmonics, the compression ratio is approximately 400 to 1 (assuming the sampling rate is 70×140). The actual compression ratio depends on the sampling rate of the lighting direction and surface properties. During relighting, we must reconstruct the reflectances from the spherical harmonic coefficients by calculating a summation of multiplications.[4] Hence, the major computation time of the software relighting is spent on



16 When there's occlusion (between the directional light source and the surface) as in (a), we might not be able to generate correct relighting for a nondirectional light (b).

reconstruction. We can achieve further reduction by exploiting the data correlation between adjacent pixels.

Our recent result shows that the coefficient vectors of neighboring pixels are similar in values. By applying vector quantization on the spherical harmonic coefficient vectors, we can further improve the overall compression ratio to approximately 1,800 to 1 (assuming the sampling rate is 70×140). A 1024×256 cylindrical panoramic image with illumination information incorporated requires only 3 to 4 Mbytes to store, which is reasonable for current computer storage and network bandwidth.

Limitations

One limitation of the proposed representation is that we can't correctly account for global illumination. The relighting equation (Equation 3) assumes the only contribution to the reflected radiance through a pixel window results from a directional light source. If part of the reflected radiance is due to interreflection, it's impossible to figure out because we don't have the detail scene geometry. Hence, we can't correctly simulate global illumination because of the lack of geometry.

Another limitation is shadowing. Although the representation can correctly address shadowing introduced by directional light sources, we might not correctly account for shadowing when using a nondirectional light source.

Figure 16a illustrates the perspective snapshot of a scene from a panoramic image's fixed viewpoint. A shadow is cast by the rectangular box onto the sphere during scene capture since the directional light source is infinitely far away. During relighting, the result will be incorrect if a point light source is positioned in between the two objects. Our approach will not generate the correct image (see Figure 16b). Instead, there will be an incorrect shadow on the sphere, because the relighting algorithm assumes that there's no occlusion between the light source and the illuminated surface that's visible through the pixel window. If occlusion exists, as in Figure 16a, that assumption is violated and hence relighting is incorrect.

Future direction

One future direction is to generalize the representation to concentric mosaics,[12] which lets the viewpoint move within a circular region.

Readers can find demonstrative movies and a prototype viewer at http://www.cse.cuhk.edu.hk/~ttwong/papers/pano/pano.html. You can also find sample movies on *CG&A*'s Web site by visiting http://computer.org/cga/cg2001/g2toc.htm (see the "Web Extras" sidebar). ■

Web Extras

The following movies demonstrate the result of relighting panorama with different types of light sources.

- Sample 1: A movie showing the screen capture of the perspective display of an indoor panorama, attic. A directional light source relights the image-based scene: http://computer.org/cga/cg2001/extras/g2032x1.mpg.
- Sample 2: A spotlight and a slide projector light relighting the same attic scene: http://computer.org/cga/cg2001/extras/ g2032x2.mpg.
- Sample 3: This movie shows the screen capture of relighting a panorama of an outdoor environment. It mimics the natural illumination throughout the whole day: http://computer.org/cga/cg2001/extras/g2032x3.mpg.

Acknowledgments

We thank the anonymous reviewers for their constructive comments. The Research Grants Council of the Hong Kong Special Administrative Region, under RGC Earmarked Grants (Project No. CUHK 4186/00E) and RGC Co-operative Research Centres (CRC) Scheme (Project No. CRC 4/98), supports this project.

References

1. J.S. Nimeroff, E. Simoncelli, and J. Dorsey, "Efficient Rerendering of Naturally Illuminated Environments," *Fifth*

Eurographics Workshop Rendering, Springer-Verlag, New York, 1994, pp. 359-373.
2. P.N. Belhumeur and D.J. Kriegman, "What Is the Set of Images of an Object Under All Possible Lighting Conditions," *IEEE Conf. Computer Vision and Pattern Recognition*, IEEE Computer Soc. Press, Los Alamitos, Calif., 1996, pp. 52-58.
3. G. Golub and C. van Loan, *Matrix Computations*, Johns Hopkins Univ. Press, Baltimore, Md., 1989.
4. T. Wong et al., "Image-based Rendering with Controllable Illumination," *Eighth Eurographics Workshop Rendering*, Springer-Verlag, New York, 1997, pp. 13-22.
5. M. Levoy and P. Hanrahan, "Light Field Rendering," *Computer Graphics Proc.* (Siggraph 96), ACM Press, New York, 1996, pp. 31-42.
6. S.J. Gortler et al., "The Lumigraph," *Computer Graphics Proc.* (Siggraph 96), ACM Press, New York, 1996, pp. 43-54.
7. Y. Yu and J. Malik, "Recovering Photometric Properties of Architectural Scenes from Photographs," *Computer Graphics Proc.* (Siggraph 98), ACM Press, New York, 1998, pp. 207-217.
8. P.E. Debevec and J. Malik, "Recovering High Dynamic Range Radiance Maps from Photographs," *Computer Graphics Proc.* (Siggraph 97), ACM Press, New York, 1997, pp. 369-378.
9. I.W. Busbridge, *The Mathematics of Radiative Transfer*, Cambridge Univ. Press, Cambridge, Mass., 1960.
10. R. Courant and D. Hilbert, *Methods of Mathematical Physics*, Interscience, New York, 1953.
11. F.X. Sillion et al., "A Global Illumination Solution for General Reflectance Distributions," *Computer Graphics Proc.* (Siggraph 91), ACM Press, New York, 1991, pp. 187-196.
12. H. Shum and L. He, "Rendering with Concentric Mosaics," *Computer Graphics Proc.* (Siggraph 99), ACM Press, New York, 1999, pp. 299-306.

Tien-Tsin Wong *is an assistant professor in the Department of Computer Science and Engineering at the Chinese University of Hong Kong. His research interests include image-based modeling and rendering, medical visualization, natural phenomena modeling, and photorealistic and nonphotorealistic rendering. He has a BSc, MPhil, and PhD in computer science from the Chinese University of Hong Kong.*

Pheng-Ann Heng *is an associate professor at the Department of Computer Science and Engineering and the director of the Virtual Reality, Visualization, and Imaging Research Center at the Chinese University of Hong Kong. His research interests include virtual reality applications in medicine, scientific visualization, 3D medical imaging, user interfaces, rendering and modeling, interactive graphics, and animation. He has a BSc in computer science from the National University of Singapore and an MSc in computer science, MA in applied math, and a PhD in computer science from the Indiana University.*

Chi-Wing Fu *is a PhD student in computer science at Indiana University, Bloomington. He has a MPhil from the Chinese University of Hong Kong. His research interests are in image-based modeling and rendering and astronomical visualization of large-scale astrophysics data.*

Readers may contact Wong at the Dept. of Computer Science and Eng., Chinese Univ. of Hong Kong, Shatin, Hong Kong, email ttwong@acm.org, http://www.cse.cuhk.edu.hk/~ttwong.