Non-Iterative, Robust Monte Carlo Noise Reduction

Ruifeng Xu, Sumanta N. Pattanaik, *Member, IEEE*

Abstract— A novel Monte Carlo noise reduction operator is proposed in this paper. We apply and extend the standard bilateral filtering method and build a new local adaptive noise reduction kernel. It first computes an initial estimate for the value of each pixel, and then applies bilateral filtering using this initial estimate in its range filter kernel. It is simple both in formulation and implementation. The new operator is robust and fast in the sense that it can suppress the outliers, as well as the inter-pixel incoherence in a non-iterative way. It can be easily integrated into existing rendering systems as a post-processing step. The results of our approach are compared with those of other methods. A GPU implementation of our algorithm runs in 500ms for a 512×512 image.

Index Terms— Monte Carlo method, noise reduction, bilateral filtering, global illumination, image processing.

1 Introduction

Computing accurate global illumination is one of the most challenging tasks in the computer graphics field. Its difficulty comes from the complexity of the rendering equation [Kajiya 1986]. Monte Carlo method is a major method used to solve this equation. Although it is powerful enough to compute all global illumination effects, too much rendering time is required to accurately render a typical scene. Limited rendering time often brings about a particular artifact in the final rendered image, known as Monte Carlo noise.

An alternative way for high quality Monte Carlo rendering is to render images at low sampling density, and then denoise them in a post-processing stage. Although a classic image noise reduction algorithm can be used to reduce this particular type of noise, a naive application is not always very effective. So, a lot of work has been devoted to carry out effective and efficient post-processing of Monte Carlo noise [Lee 1990; Rushmeier 1994; Jensen 1995; Tamstorf 1997; McCool 1999]. A comprehensive account of this literature is given in [McCool 1999].

Rushmeier et al. [1994] considered the Monte Carlo noise as "*outliers*", and used an energy-preserving non-linear filter to suppress these outliers. Jensen et al. [1995] found that most Monte Carlo noise appeared in images as inter-pixel incoherence, and attempted to reduce it by using classical image denoising algorithms, like median filtering.

Of the noise removal techniques, anisotropic diffusion [McCool 1999] is the most relevant to our work. It reduces the inter-pixel incoherence by applying an anisotropic diffusion algorithm to the pixel value. The denoising results look impressing. But, this approach is sensitive to outliers. It is also very slow because of its iterative nature.

It is now clear that Monte Carlo noise appears both as outliers and as inter-pixel incoherence in a typical image rendered at low sampling density [Jensen 1995; Lee 1990; McCool 1999; Rushmeier 1994]. Unfortunately, none of previous approaches can reduce both types of noise in a unified way. In this paper, we propose such a unified Monte Carlo noise reduction approach to suppress both outliers and inter-pixel incoherence, using *bilateral filtering* [Tomasi 1998].

Inspiration and Our Contribution

Our work is inspired by the work of Tomasi et al. [1998], where bilateral filtering is proposed to smooth images while keeping the edges undisturbed. Bilateral filtering is also successfully applied to image denoising [Elad 2002a; Elad 2002b], mesh smoothing and denoising [Jones 2003; Fleishman 2003], and high dynamic range tone mapping [Durand 2002]. A theoretical analysis of this technique is presented in [Barash 2001]. The principle of bilateral filtering is simple. It combines the domain filtering and range filtering, as shown in Equation 1.

$$h(x)=\frac{\int_{-\infty}^{\infty}\int_{-\infty}^{\infty} f(\xi)c(\xi,x)s\big(f(\xi),f(x)\big)d\xi}{\int_{-\infty}^{\infty}\int_{-\infty}^{\infty} c(\xi,x)s\big(f(\xi),f(x)\big)d\xi} \tag{1}$$

where, $h(x)$ is the estimator of the current pixel x, $f(x)$ is the pixel value of x and $f(\xi)$ is the pixel value of its neighbor(s) ξ, and $s\big(f(\xi),f(x)\big)$ and $c(\xi,x)$ are the range filter and domain filter kernels. They are often modeled as Gaussian functions with parameters σ_r,σ_d respectively, as shown in Equation 2.

$$c(\xi,x)=\exp\left\{-\frac{1}{2}\left(\frac{|\xi-x|}{\sigma_d}\right)^2\right\}$$
$$s\big(f(\xi),f(x)\big)=\exp\left\{-\frac{1}{2}\left(\frac{|f(\xi),f(x)|}{\sigma_r}\right)^2\right\} \tag{2}$$

If some neighbor ξ is an outlier, it has a much larger or much smaller value $f(\xi)$ than that of the central point x. Its contribution to the estimator $h(x)$ will be greatly reduced by the range filter $s\big(f(\xi),f(x)\big)$, which favors similar range values rather than disparate values. Bilateral filter is a robust local adaptive filter, which can be used to enhance image coherence. However, as illustrated in Figure 1, it cannot be directly used to suppress the outliers of Monte Carlo noise. And in the next section, we show that the original bilateral filtering is not

- *Ruifung Xu is with School of Computer Science, UCF, Orlando, Fl-32816. E-mail: rxu@cs.ucf.edu*
- *Sumanta N. Pattanaik is with School of Computer Science, UCF, Orlando, Fl-32816. E-mail: sumant@cs.ucf.edu.*

Manuscript revisions: First revision 1/29/2004, Second revision 2/20/2004.

as robust as claimed in [Durand 2002; Jones 2003]. We extend the standard bilateral filtering to handle outliers and inter-pixel incoherence in a unified framework. Our contributions are:

- Application of bilateral filtering to Monte Carlo noise reduction.
- Extension of bilateral filtering with an initial estimation preprocess.

The rest of the paper is organized as follows. Section 2 presents our Monte Carlo noise operator developed from bilateral filter. Section 3 describes a denoising framework, which can be easily integrated into existing rendering system. Experimental results and analysis are given in the last two sections.

2 MONTE CARLO NOISE REDUCTION OPERATOR

The outliers in Monte Carlo noise are pixels with much larger or much smaller values than its neighbors. It is desirable to remove them together with inter-pixel incoherence while keeping edges undisturbed. A Gaussian filter blurs the edges. Therefore, McCool et al [1999] introduced anisotropic diffusion to suppress inter-pixel incoherence while keeping edges intact. Standard bilateral filtering can do the same thing as anisotropic diffusion, but neither of them can effectively remove the outliers. This is because the initial estimator $f(x)$ used in $s(f(\xi), f(x))$ is far different from its true value, and very little contribution to $h(x)$ comes from such neighbors due to the infinitesimal weights returned by the range function. Thus, the outliers remain almost unchanged. They are neither suppressed, nor do they contribute to their neighbors. As shown in Figures 1(b) and 1(c), the outliers remain there after applying standard bilateral filtering. Fortunately, standard bilateral filtering can be extended to suppress both outliers and inter-pixel incoherence while keeping edges intact. We propose to employ an initial near-true estimator $\tilde{f}(x)$ to replace $f(x)$, and make use of Equation 3 as our new Monte Carlo noise reduction operator. Note that we use $\hat{f}(x)$ to replace $h(x)$ to denote the new estimator using bilateral filtering around point x.

$$\hat{f}(x) = \frac{\int_{-\infty}^{\infty}\int_{-\infty}^{\infty} f(\xi)c(\xi,x)s\left(f(\xi), \tilde{f}(x)\right)d\xi}{\int_{-\infty}^{\infty}\int_{-\infty}^{\infty} c(\xi,x)s\left(f(\xi), \tilde{f}(x)\right)d\xi} \tag{3}$$

There are various possible options for $\tilde{f}(x)$, such as mean value around pixel x, or median value around pixel x. From our experiments, we find that Gaussian filtered value (shown in Equation 4) performs the best in dealing with Monte Carlo noise.

$$\tilde{f}(x) = \frac{\int_{-\infty}^{\infty}\int_{-\infty}^{\infty} f(\xi)c(\xi,x)d\xi}{\int_{-\infty}^{\infty}\int_{-\infty}^{\infty} c(\xi,x)d\xi} \tag{4}$$

Figure 1 shows the denoising of "*living room*" (see http://radsite.lbl.gov/) using original bilateral filtering, iterative bilateral filtering, and our bilateral filtering extension. The Gaussian parameters used in all the cases are the same: $\sigma_r = 2.0, \sigma_d = 0.4$. Standard bilateral filtering (Figure 1(b)) is almost ineffective in reducing Monte Carlo noise. In Figure 1(c), the bilateral filtering is iterated 20 times [Elad 2002a], and the incoherence inside regions are well suppressed, but the outliers remains unchanged. Notice the outliers on the window of Figure 1(c). It shows that the standard bilateral filtering is not so robust in suppressing outliers!

(a) (b) (c) (d)

Figure 1. Outliers Reduction using Bilateral Filtering. (a) Noisy image; (b) Standard bilateral filtering; (c) Iterative bilateral filtering used in [Elad 2002] 20 iterations. (d) Our new bilateral filtering operator.

Figure 1(d) shows the success of our extension to bilateral filtering. Our Monte Carlo noise reduction operator can also be used to reduce noises other than Monte Carlo noise.

Numerical formulation

The integration in Equations 3 and 4 are evaluated as discrete summation. As the weight function is very small at a distance farther than $3\sigma_d$ away from the central pixel ($e^{-(3\sigma_d)^2/(2(\sigma_d)^2)} = e^{-9/2} < 0.012$), we select a square window around the current pixel of size $6\sigma_d \times 6\sigma_d$ as the neighborhood window. The discrete version of the equations in this window is shown in Equations 5 and 6.

$$\hat{f}(i,j) = \frac{\sum_{u=-3\sigma_d}^{3\sigma_d}\sum_{v=-3\sigma_d}^{3\sigma_d} f(i+u,j+v)c(u,v)s\left(f(i+u,j+v),\tilde{f}(i,j)\right)}{\sum_{u=-3\sigma_d}^{3\sigma_d}\sum_{v=-3\sigma_d}^{3\sigma_d} c(u,v)s\left(f(i+u,j+v),\tilde{f}(i,j)\right)} \quad (5)$$

where,

$$\tilde{f}(i,j) = \frac{\sum_{u=-3\sigma_d}^{3\sigma_d}\sum_{v=-3\sigma_d}^{3\sigma_d} f(i+u,j+v)c(u,v)}{\sum_{u=-3\sigma_d}^{3\sigma_d}\sum_{v=-3\sigma_d}^{3\sigma_d} c(u,v)} \quad (6)$$

Our computation first finds the initial estimated value $\tilde{f}(i,j)$ for each pixel. Then, a bilateral filtering step is executed using $\tilde{f}(i,j)$. It is a non-iterative process, and the computation is fast. The denoising effects are greatly enhanced by this single additional initial estimation step, as shown in Figure 1(d). The pseudocode is briefly described in Figure 3. The whole process is a loop over each pixel *p* in the image. Inside the loop first the range filter kernel is constructed using parameter σ_r and initial estimator $\tilde{f}$ and then is combined with the domain filter kernel. The resulting bilateral filter kernel is centered on *p* and integrated with the pixels within the square window of size $6\sigma_d \times 6\sigma_d$ to get the filtered value for the pixel *p*.

3 Denoising Framework

As described by Jensen [Jensen 1995], most of the noise arises from computing diffuse inter-reflection (*indirect component*) using Monte Carlo methods. The contribution from direct illumination and specular inter-reflection (*direct component*) carries little noise. We follow this observation, and denoise only the indirect component. The denoised indirect component is then added to the direct component to get the final denoising result. The direct and indirect components are easily separated by adding only a few lines of code into the Monte Carlo renderer. The whole denoising process is briefly shown in Figure 2.

Our denoising framework can be easily integrated to the Monte Carlo rendering pipeline as a post-processing stage. The indirect and direct components are outputs of rendering processes. After denoising (see Figure 3 for an overview of the denoising algorithm), it is sent to other stages for further processing, like tone mapping, data compression, etc. With our denoising technique, a Monte Carlo renderer can use low sampling rates for higher quality image in shorter time.

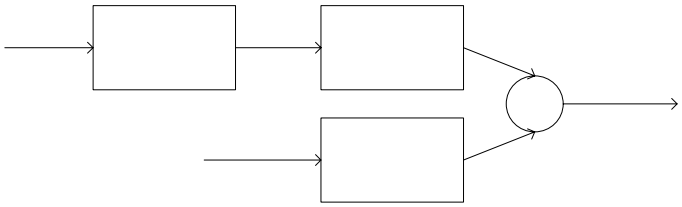


Figure 2: Our denoising Framework

```
Algorithm MC-denoising
Construct domain filter kernel c with σ_d;
f̃ = I * c; /*convolution for initial estima-
      tor*/
For each pixel p
  Construct range filter kernel s with σ_r
    and f̃;
  κ = c·s; /*combine domain and range fil-
        ters*/
  κ = κ/|κ|; /*normalization*/
  f̂ = I(p) * κ;
  Set the value at pixel p with estimator f̂;
```

Figure 3: Pseudocode of our algorithm.

4 Experimental Results

We have implemented our denoising algorithm in C. The direct and indirect components are obtained by adding several lines of code to "*rpict*" in Radiance (see http://radsite.lbl.gov/) to save the direct and indirect components separately.

Monte Carlo noise has several ways to contaminate the pixel color, e.g., hue and luminance. Following the approach of [Rushmeier 1994] and [McCool 1999], we assume that luminance channel is most likely contaminated. We compute the luminance for each pixel using Equation 7 [Ward 1996].

$$I(R,G,B) = 0.265*R + 0.670*G + 0.065*B \quad (7)$$

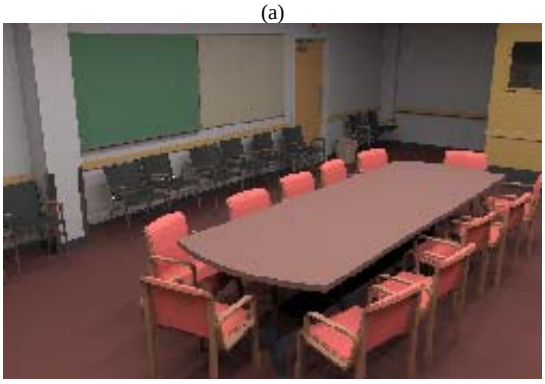
Our denoising results also show that luminance carries most Monte Carlo noise. It is worth mentioning that we carry out Monte Carlo noise reduction in the logarithm domain of the luminance channel. This is because the human eye has a linear response to the logarithm of pixel luminance value.

Figures 4 and 5 show two denoising examples using our Monte Carlo noise reduction operator. More explanation can be found in the caption of Figure 5.

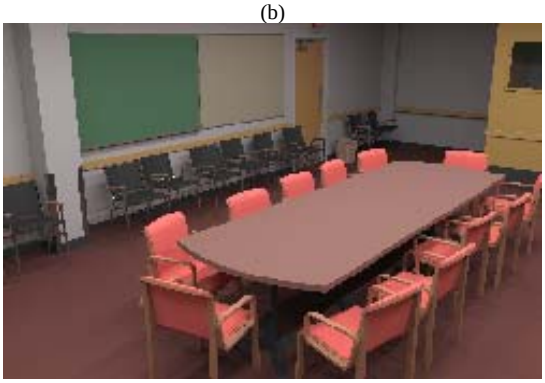
Figure 6 lists time to generate the images in Figures 1, 4 and 5. Our experimental platform is a Celeron 2.0GHz (392M memory, Windows2000). The numbers in parentheses are the number of samples per pixel (sampling rate). In each cell of column 2 and 3, the numbers on the second line are the mean square error (MSE). The denoising time is only a small fraction of the noisy image rendering time, and the time complexity of our denoising algorithm is *O(n)* in most cases, where *n* is pixel number of the noisy image. We can see that a Monte Carlo renderer using our noise reduction method produces higher quality images in shorter time as compared to producing an image of the same quality by merely improving the sampling rate.

The C source code and executable are available as http://www.cs.ucf.edu/~rxu/mcnr/mcnrBiFilter.c and http://www.cs.ucf.edu/~rxu/mcnr/mcnrBiFilter.exe.

(a)



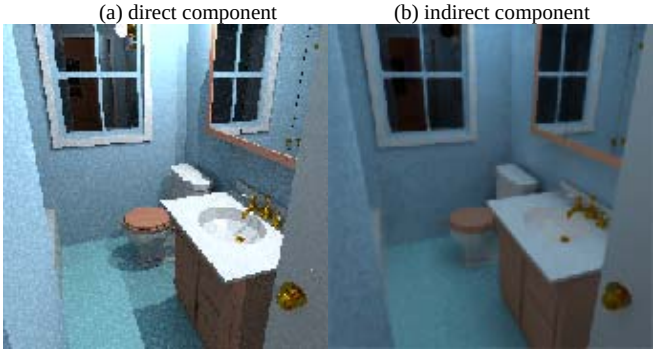
(b)



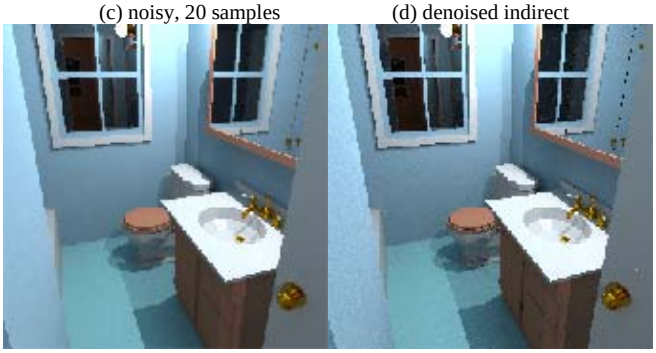
(c)

Figure 4: Denoising of "*conference room*" image. (b) is the denoised result of the image in (a) generated using 5 samples per pixel. This image is very similar to the accurate result shown in (c) which is obtained using 400 samples per pixel.

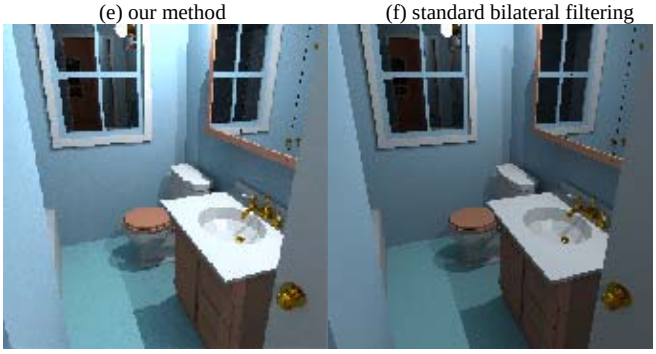
(a) direct component (b) indirect component



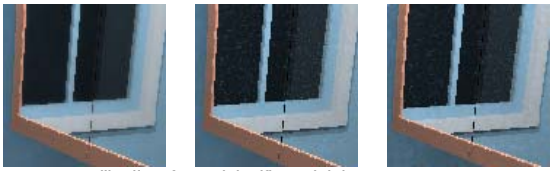
(c) noisy, 20 samples (d) denoised indirect



(e) our method (f) standard bilateral filtering



(g) Wiener filtering (h) accurate, 300 samples



(i) clips from (e), (f) and (g)

Figure 5: Some results of our Monte Carlo noise reduction operator on "cabin". (a)-(i) shows the whole denoising process for "*cabin"* image. (a) and (b) are the direct and indirect components of (c). (b) is denoised using our method to obtain (d). And (e) is the denoising result by adding up (a) and (d). For comparison, we also show the denoising results (f) using standard bilateral filtering, (g) using *Wiener* filtering. (h) is the accurate image obtained using 300 samples (setting *ad=300* in "*rpict*"). (i) shows clips of the right window on images (e),(f),(g), from left to right. It is apparent that the outliers are removed in (e), but remain in (f), (g). The models of the scene used to generate the images are courtesy of Ward (see http://radsite.lbl.gov/)

	noisy	denoised	accurate	σ_r, σ_d
Living room 400×300	50s(2) 0.152	5.0s 0.089	2100s (500)	2, 0.4
Cabin 512×512	286s(20) .3630	9.8s .2202	3602s (300)	2, 0.4
Conf. room 512×347	183s(5) .0312	6.5s .0275	1802s (400)	2, 0.4

Figure 6: Statistics data. (Note: numbers in parentheses denote the sampling rate.)

We use MSE as a simple fidelity metric to confirm the relative image quality improvement between the coarsely rendered image and the denoised image. The comparison basis is the image for the same view rendered at very high quality. The MSE measurement is performed using the logarithm of the luminance value, as shown in Equation 7. The larger the MSE, the noisier the image. For the "conference room" in Figure 4, MSE of (a) and (b) with respect to (c) is 0.0312 and 0.0275, respectively. And in Figure 5, MSE of (c) and (e) with respect to (h) is 0.3630 and 0.2202, respectively. Our denoising algorithm does improve the image quality by reducing the MSE.

Parameters Setting

There are two parameters involved in our algorithm: σ_r, σ_d for range and domain filters. In spite of the efforts by Jones et al. [2003], automatic estimation of the bilateral filter parameters remains an open problem. Fortunately, we find $\sigma_r = 2, \sigma_d = 0.4$ are appropriate for most cases of Monte Carlo noise reduction, and we used $\sigma_r = 2, \sigma_d = 0.4$ in all of our experiments in this paper.

Although these parameters are only established through experiments, we believe they are closely related to some aspects of human perception including spatial vision and color discrimination.

5 Conclusions and Future Work

We propose a non-iterative local adaptive filter based on bilateral filter for Monte Carlo noise reduction. Unlike other Monte Carlo reduction methods, our approach is able to suppress outliers and inter-pixel incoherence in a unified framework. It can also be used in other denoising tasks, like mesh denoising, where outliers and inter-pixel incoherence coexist. A standard bilateral filtering may be enough in cases where only inter-pixel incoherence needs to be reduced.

The strength of our method lies in its simplicity, robustness and efficiency. It reduces both types of noise in only two passes. The method can be easily adapted to parallel implementation, as well as GPU implementation. We implemented the latter on an ATI RADEON 9700 graphics card which executes the denoising of 512×512 image in a fraction of second. For the "*cabin*" image in Figure 5, our GPU implementation runs at about 2 fps.

This method requires only two parameters σ_r, σ_d. Although further tuning is possible $\sigma_r = 2, \sigma_d = 0.4$ can be used in most cases of Monte Carlo noise reduction. Automatic setting of these parameters is one future research topic.

"Robustness" of our approach lies in the fact that it can effectively suppress Monte Carlo noise in presence of ourliers. It can very well suppress inter-pixel incoherence and outliers together.

Acknowledgment

This work was supported in part by grants from the Office of Naval Research, Florida High Tech Corridor (FHTC-Phase VII) and ATI Technologies, Inc.

References

[1] Barash, D. 2001. Bilateral Filtering and Anisotropic Diffusion: Towards a Unified Viewpoint. In *Third International Conference on Scale-Space and Morphology,* 273-280.

[2] Durand, F. and Dorsey, J. 2002. Fast Bilateral Filtering for the Display of High-Dynamic-Range Images. In *Proceedings of ACM SIGGRAPH 2002*, 257-266.

[3] Elad, M. 2002. Analysis of the Bilateral Filter. In *The 36th Asilomar on Signals, Systems and Computers*, Pacific Grove, CA.

[4] Elad, M. 2002. On the Origin of the Bilateral Filter and Ways to Improve It. In *IEEE Trans. On Image Processing,* 11(10), 1141-1151.

[5] Fleishman, S., Drori, I., and Cohen-Or, D. 2003. Bilateral Mesh Filtering. In *Proceedings of ACM SIGGRAPH 2003*, 950-953, San Diego, TX.

[6] Jensen, H. W., and Christensen, N. J. 1995. Optimizing Path Tracing using Noise Reduction Filters. In *Proceedings of the WSCG 1995*, 134-142.

[7] Jones, T. R., Durand, F., and Desbrun, M. 2003. Non-Iterative, Feature-Preserving Mesh Smoothing. In *Proceedings of ACM SIGGRAPH 2003*, 943-949, San Diego, TX.

[8] Kajiya, J. The Rendering Equation. 1986. In *Proceedings of ACM SIGGRAPH 86*, 143-150.

[9] Lee, M.E., and Redner, R. A. 1990. Filtering: A Note on the Use of Nonlinear Filtering in Computer Graphics. In *Computer Graphics*, 10(3), 23-29.

[10] McCool, M. D. 1999. Anisotropic Diffusion for Monte Carlo Noise Reduction. In *ACM Trans. On Graphics*, 18(2), 171-194.

[11] Purgathofer, W. 1987. A Statistical Method for Adaptive Stochastic Sampling. In *Computers and Graphics*, 11(2), 157-162.

[12] Rushmeier, H. E., and Ward, G. J. 1994. Energy Preserving Nonlinear filters. In *Proceedings of ACM SIGGRAPH 94*, 131-138.

[13] Simoncelli, E. P. 1999. Bayesian Denoising of Visual Images in the Wavelet Domain. In *Bayesian Inference in Wavelet Based Models,* eds Muller P. and Vidakovic, B., 291-308, Springer-Verlag, New York.

[14] Tamstorf, R., and Jensen, H. W. 1997. Adaptive Sampling and Bias Estimation in Path Tracing. In *Rendering Techniques '97*, 285-295. Eds. Dorsey, J. and Slusallek, P. H.

[15] Tomasi, C., and Manduchi, R. 1998. Bilateral Filtering for Gray and Color Images. In *Proceedings of the IEEE ICCV 1998*, 839-846, Bombay, India.

[16] Ward, G. 1996. Radiance E-mail Archive: http://radsite.lbl.gov/radiance/digests_html/v3n0.html.

Ruifeng Xu received his Masters degree in Computer Science from Zhejiang University. Since August 2001, he is a Ph.D. student in the School of Computer Science, UCF. His current research interests include global illumination, realistic rendering in real-time, and high dynamic range video compression.

Sumanta N. Pattanaik received his Ph.D. degree in Computer Science in 1993 from BITS-Pilani, India. He was a researcher in IRISA/INRIA, France from 1993 to 1995 and in Program of Computer Graphics at Cornell University from 1995 to 2001. Since July 2001, he is an associate professor of Computer Science at University of Central Florida. He is a Category Editor (Computer Graphics) of ACM Computing Reviews. His current research interest is real-time realistic rendering using programmable graphics hardware. He is a member of IEEE.