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 the initial estimate of each pixel, and then applies bilateral filtering using this initial estimate in its range filtering 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, and such a framework is shown in this paper. 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 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 1998], where Bilateral Filtering is proposed to filter gray and color images. It has the property of smoothing images while keeping the edges undisturbed. [Durand 2002] and [Jones 2003] related it to *Robust Statistics*, and built an estimator using bilateral filter, which is expected to be robust in the presence of outliers or larger deviation from a theoretical distribution. Bilateral filtering is already successfully applied to image filtering [Tomasi 1998], image denoising [Elad 2002a; Elad 2002b], mesh smoothing and denoising [Jones 2003; Fleishman 2003]. A theoretic analysis 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.

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

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

$$c(\xi,x)=\exp\left\{-\frac{1}{2}\left(\frac{|\xi-x|}{\sigma_d}\right)^2\right\} \quad (2)$$

$$s(f_\xi, f_x)=\exp\left\{-\frac{1}{2}\left(\frac{|f_\xi-f_x|}{\sigma_r}\right)^2\right\}$$

If some neighbor ξ is an outlier, it has a much larger or much smaller value f_ξ than that of the central point x. Its contribution to the estimator $h(x)$ will be greatly reduced by the range filter $s(f_\xi, f_x)$, 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

- *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 revised (1/29/2004).

illustrated in Figure 2, 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 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 filtering. 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 MONTECARLO NOISE REDUCTION OPERATOR

The outliers in Monte Carlo noise are singular pixels with much larger or much smaller values than its neighbors. It is desirable to remove them together with intra-region incoherence while keeping edges undisturbed. A Gaussian filter will blur the edges, and therefore anisotropic diffusion is introduced to suppress intra-region incoherence while keeping edges intact [McCool 1999]. Standard bilateral filter can do the same thing as anisotropic diffusion, but it, as well as anisotropic diffusion, can't effectively remove the outliers, as shown in Figure 2. 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 its estimator comes from such neighbors due to the infinitesimal weights returned by the range function. Thus, the outliers remain almost unchanged. And they are neither suppressed, nor do they contribute to their neighbors. The outliers will remain there after applying standard bilateral filtering, as shown in Figures 2(b), (c). Fortunately, standard bilateral filter is ready to be extended to suppress both outliers and intra-region 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 we use $\hat{f}(x)$ to replace $h(x)$ for convenience, denoting the new estimator using bilateral filtering around point x.

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

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

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

Our experimental results in Figure 2 show that results using our extension to bilateral filtering. Figure 2 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 2(b)) is almost ineffective in reducing Monte Carlo noise. In Figure 2(c), the bilateral filtering is iterated 20 times [Elad 2002a], and the incoherence inside regions are well suppressed, but the outliers remains unchanged.

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

Figure 2. 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.

Notice the outliers on the window of Figure 2(c). No matter how many bilateral filtering iterations are applied, they never

disappear. It shows that the standard bilateral filter is not so robust in suppressing outliers! Figure 2(d) shows the success of our extension to bilateral filtering. Our Monte Carlo noise reduction operator can also be used to reduce other types of noise than Monte Carlo noise.

Numerical formulation

Equations 3 and 4 are evaluated numerically. As the weight function is very small when 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 with 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.

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

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 a single additional initial estimate step, as shown Figure 2(d). The pseudocode can be briefly described in Figure 3. The whole process is a loop over each pixel, where the range filter is first contructed using parameter σ_r and initial estimator $\tilde{f}$, and then is convolved with original image $I(p)$ and domain fitler to obtain the bilateral filter estimator $\hat{f}$.

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 for the final denoising result. The direct and indirect components are easily separated, by adding only a few lines into the Monte Carlo renderer. The whole denoising process is briefly shown in Figure 4.

Our denoising framework can be easily integrated to the 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 algorithn), it is sent to other stages for further processing, like tone mapping, storage, or displaying. With our denoising technique, the Monte Carlo renderer can use low sampling rates for fast quality image.

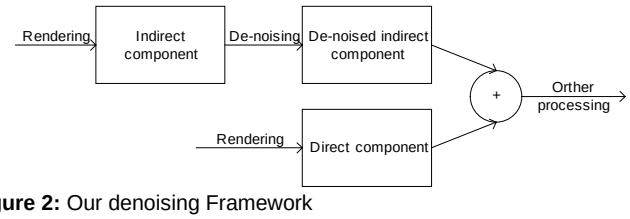


Figure 2: Our denoising Framework

Algorithm MC-denoising
Construct domain filtering kernel c with σ_d;
$\tilde{f} = I * c$ /*convolution for initial estimator*/
For each pixel p
  Construct range filtering kernel s with $\sigma_r, \tilde{f}$;
  $\kappa = c * s$
    /*combine domain and range filters*/
  $\kappa = \kappa / |\kappa|$ /*normalization*/
  $\hat{f} = I(p) * \kappa$;
  Set pixel p with estimator $\hat{f}$;

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.

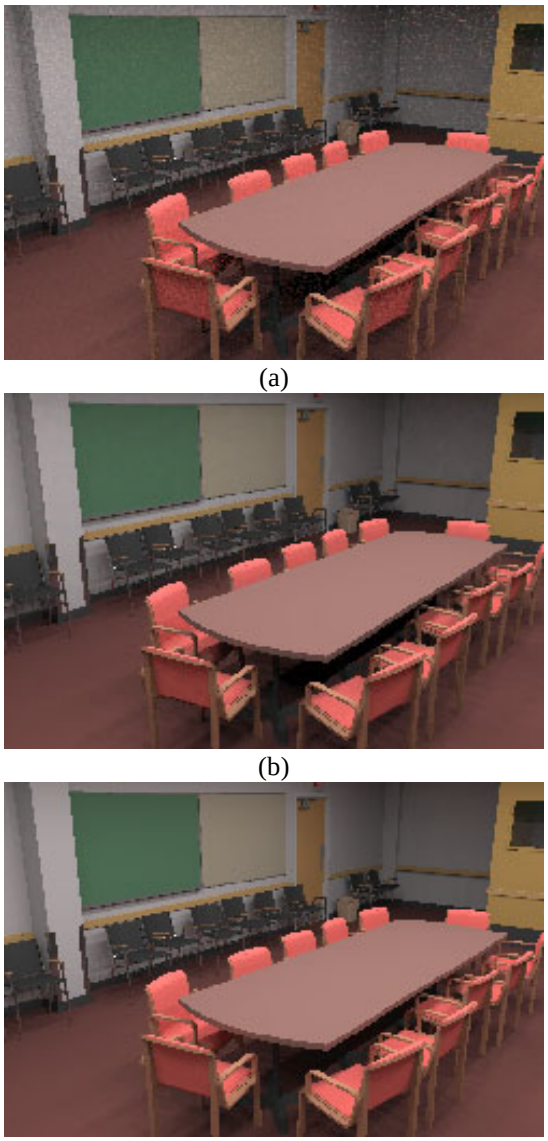
Monte Carlo noise has several ways to contaminate the pixel color, e.g., hue and luminance. We take the approach followed by [Rushmeier 1994] and [McCool 1999], i.e., luminance is most likely contaminated. We use the luminance computation formula in Radiance [Ward 1996], as shown in Equation 7.

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

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.

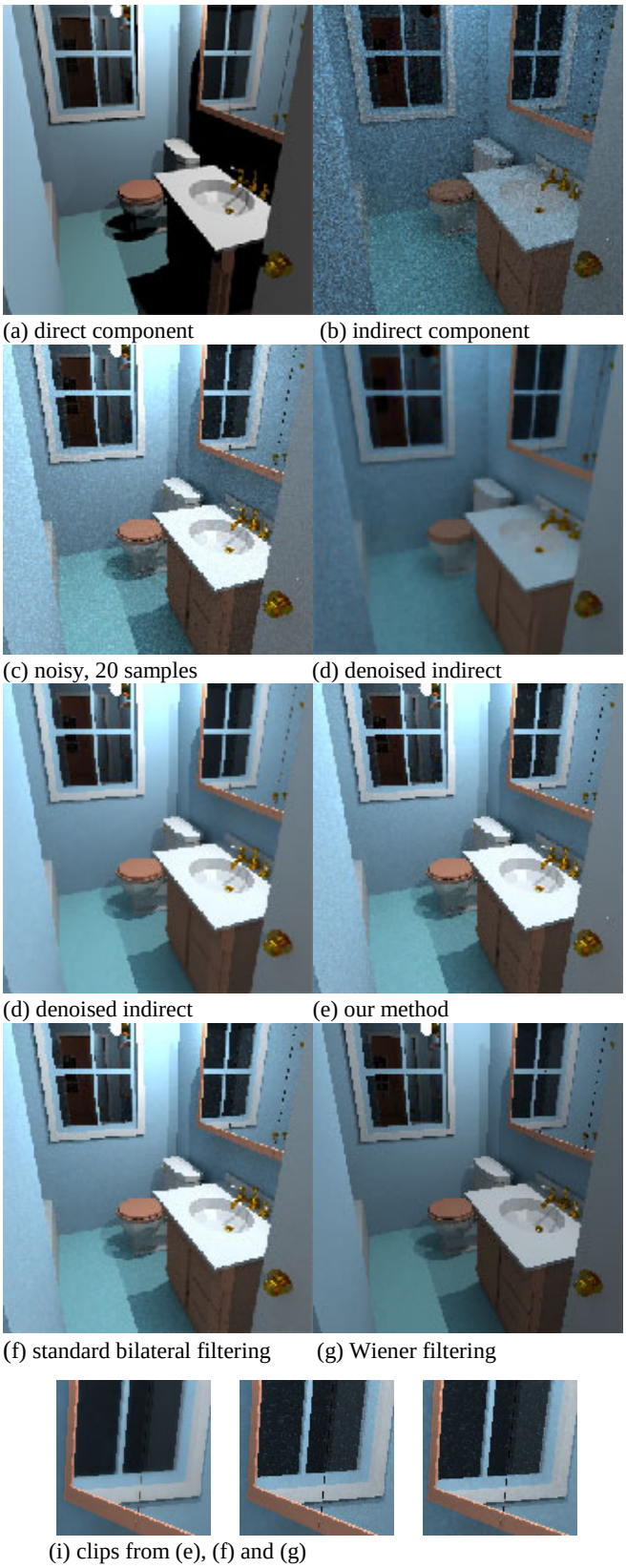
Figures 5 and 6 show two denoising examples using our Monte Carlo noise reduction operator. More explaination can be found in the title of Figure 6.

Figure 7 lists time to generate the images in Figures 2,5 and 6. Our experimental platform is a Celeron 2.0GHz (392M memory, Windows2000). The numbers in parentheses are sampling rate. In each cell of column 2 and 3, the numbers on the second line are the 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 much time is saved for quality rendered images using our Monte Carlo noise reduction method, compared to merely improving sampling rate. The C source code and executable are available at http://www.cs.ucf.edu/~rxu/mcnr/mcnrBiFilter.c and http://www.cs.ucf.edu/~rxu/mcnr/mcnrBiFilter.exe.



(a) (b) (c)

Figure 5: Denoising of "*conference room*" image. (b) is the denoised result of (a) (5 samples), which is very similar to the accurate result (c)(400 samples).



(a) direct component (b) indirect component

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

(d) denoised indirect (e) our method

(f) standard bilateral filtering (g) Wiener filtering

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

Figure 6: 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 result (f) using standard bilateral filtering, (g) using *Wiener* filtering, and the accurate image using 300 samples (setting *ad=300* in "*rpict*"). (i) shows clips of the right window on image (e),(f),(g), from top to bottom. It is apparent that the outliers are removed in (e), but remains 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 7: Statistics data. **(**Note: numbers in parentheses denote the sampling rate.)

We use mean least square (MSE) as a simple fidelity metric to confirm the relative image quality between coarsely rendered image and denoised image. The comparision basis is the same image 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 5, MSE of (a) and (b) with respect to (c) is 0.0312 and 0.0275, respectively. And in Figure 6, 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 domain and range filters. Automatic estimation of these bilateral filtering parameters remains a problem, although there are some related efforts toward this problem [Jones 2003]. 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 testified through experiments, we believe they are closely related to some aspects of human perception including spatial vision and color discrimination. A theoretical derivation is possible and we leave it for further study.

5 Conclusions and Future Work

We propose a non-iterative local adaptive filter based on bilateral filtering 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 inside-region incoherence coexist. A standard bilateral filtering is enough in cases where only inside-region incoherence needs to be reduced.

The strengths of our method lie 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 stream processor implementation. We implemented the latter on an ATI RADEON 9700 graphics card, which can execute the denoising in real-time. For the "*cabin*" image in Figure 5, our GPU implementation runs at no less than 2 fps.

This method requires only two parameters, and $\sigma_r = 2, \sigma_d = 0.4$ can be used in most cases of Monte Carlo noise reduction, although further tuning is possible. Theoretical derivation of these parameters is one future research topic.

Acknowledgment

This work was supported in part by grants from the Office of Naval Research, Florida High Tech Corroidor (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 ScaleSpace 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 Tran. 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*, 2(2),157-162.

[12] Rushmeier, H., and Ward, G. 1994. Energy Preserving Non-linear 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 from so and so 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.