Eurographics Symposium on Rendering 2003, pp. 1–11
Per Christensen and Daniel Cohen-Or (Editors)

The Trilateral Filter for High Contrast Images and Meshes

Prasun Choudhury and Jack Tumblin

Department of Computer Science, Northwestern University, Evanston, IL, USA.

Abstract

We present a new, single-pass nonlinear filter for edge-preserving smoothing and visual detail removal for N dimensional signals in computer graphics, image processing and computer vision applications. Built from two modified forms of Tomasi and Manduchi's bilateral filter, the new "trilateral" filter smoothes signals towards a sharply-bounded, piecewise-linear approximation. Unlike bilateral filters or anisotropic diffusion methods that smooth towards piecewise constant solutions, the trilateral filter provides stronger noise reduction and better outlier rejection in high-gradient regions, and it mimics the edge-limited smoothing behavior of shock-forming PDEs by region finding with a fast min-max stack. Yet the trilateral filter requires only one user-set parameter, filters an input signal in a single pass, and does not use an iterative solver as required by most PDE methods. Like the bilateral filter, the trilateral filter easily extends to N-dimensional signals, yet it also offers better performance for many visual applications including appearance-preserving contrast reduction problems for digital photography and denoising polygonal meshes.

Categories and Subject Descriptors (according to ACM CCS): I.3.3 [Computer Graphics]: Picture/Image Generation: Display Algorithms; I.3.5 [Computer Graphics]: Computational Geometry and Object Modeling: Geometric algorithms, languages, and systems

1. Introduction and Related Work

Despite decades of widespread interest in the problem [5, 6, 8, 11, 18, 23, 24, 25, 31, 33, 35], simple and robust edge-preserving smoothing of visual signals has proven elusive, because the terms are ill-defined and somewhat contradictory. Edges are *perceived* discontinuities that are not always matched to a reliable mathematical discontinuity. In the past six years, several edge-preserving smoothing methods have addressed the stubborn problem of visual appearance-preserving contrast reduction [2, 8, 10, 24, 33]. The filter described in this paper was developed for this task, but we will show that it may have broader uses such as de-noising higher dimensional data and 3D meshes.

Contrast reduction, and the closely-related "tone mapping" problem long known in photography and printing, is increasingly important because the usable contrast abilities of print and electronic displays remains small, typically between about 10:1 to 30:1, but many interesting scenes contain far greater contrasts such as 11,000:1 in Figure 1. Scenes depicted in Figures 1 and 18 are usually impossible to photograph well conventionally, because no single exposure setting for a camera can capture all the visible details in both the brightest and darkest areas. As explained in a general framework for tonemapping by Tumblin and Rushmeier [32], a tone-mapped image should express exactly what a human observer would see in the original scene, including glare, afterimages, floaters, and all other human visual flaws and effects. They offered an early film-like solution, and other better methods soon followed; see [8, 18] for a detailed summary. We will not address perception, but only focus on the task of contrast reduction that preserves as much scene detail as possible without introducing objectionable artifacts.

Now that photography of even the most extreme "high dynamic range" (HDR) or high contrast scenes is now yielding to multiple-exposure image capture methods [7] (used for Figure 1) and novel cameras [19], the problem is more urgent; these new images are not directly displayable without loss of many visually important details. High contrast image display is difficult because often no one-to-one mapping of scene to display intensities is satisfactory. Photographic scaling and contrast compression (e.g. $I_{out} = M \cdot I_{in}^{\gamma}$ for $\gamma < 1$ and $0 < M < 1$) can fail because compressing large contrasts enough to match the display devices will also compress small contrasts to invisibility. Several early tone-mapping-

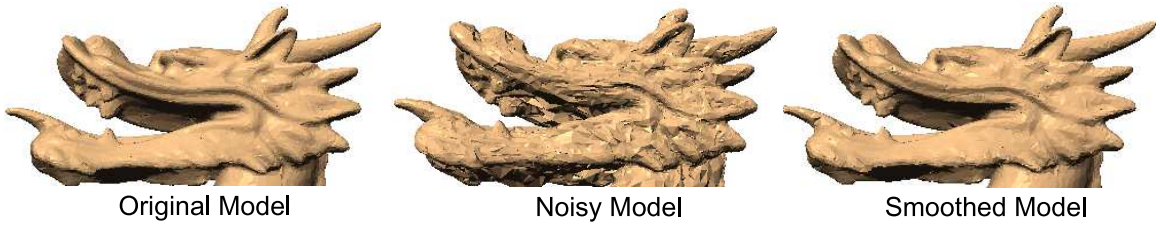

Figure 2: *Trilateral filtering can restore a noise corrupted mesh in a single pass. Middle image shows additive Gaussian noise with* $\sigma = 1/5$*th mean edge length.*

Figure 1: *Low contrast image (20:1) made with the trilateral filter from a high contrast image (11,000:1). Small images show the original scene radiances progressively scaled by factors of 10.*

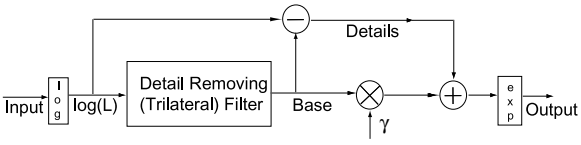

Figure 3: *Contrast reduction method based on edge-preserving filters.*

related papers [8, 18, 26] reduced contrasts by compressing only the image components selected by one or more low pass filters, but this approach can easily cause strong halo-like artifacts. However, recent papers by Ashikhmin [2] and Reinhard *et al.* [24] have largely overcome these drawbacks by selecting the best filter diameter for each pixel from an image pyramid.

Many other published detail-preserving contrast reduction methods use some form of edge-preserving smoothing [8, 10, 18, 24, 33] to separate the input image into compressible and incompressible contrast components, as shown in Figure 3. We also follow this approach. This frequently recurring contrast reduction scheme in Figure 3 is homomorphic; it filters the logarithm of intensity as Stockham did [28], using a "detail removing filter" of some sort to smooth away the small and complex variations presumably due to reflectance changes. The large simplified illumination-related filter output "Base" is compressed, usually by a scale factor γ (and sometimes offset by a scale factor $\log(M)$), and then added back to the complex details that the filter removed. Conversion from logarithmic back to a linear signal produces the displayed image result. A few papers such as [33] extended this idea by using multiple filters to refine compression amounts.

The success of the approach in Figure 3 depends entirely on the design of the "detail-removing filter." If the filter's smoothing is incomplete, the "Base" signal may destroy some important scene details due to severe contrast compression by factor γ. Worse, if the filter blurs or distorts its illumination-like output even slightly, then these distortions will escape compression as part of the "Details" signal. These errors can cause strange halo-like artifacts in the result, especially near specular highlights or in broad but strongly shaded regions such as the sky near the tree line in Figure 1. Entirely avoiding both errors continues to elude most published methods.

With few exceptions, edge-preserving filters useful for contrast reduction fall into two broad classes of (a) iterative solvers and (b) nonlinear filters. Iterative solver methods gradually and repeatedly modify an initial image I_{in} towards a final "infinite time" image I_∞ guided by a discretized partial differential equation (PDE). Research in scale-space methods using heat-flow PDEs led to anisotropic diffusion PDEs [23] that combine smoothing and edge sharpening into a single iterative process. It rapidly forms sharp, fixed shocks at edges, and gradually smoothes between them by diffusing towards a piecewise-constant I_∞ result. However, this method forms strong, spurious step-like shocks across any large high gradient region [35, 33], such as the back-lit cirrus clouds in Figure 1. To avoid this, third order curvature flow PDEs such as LCIS [33] smooth towards piecewise minimum-curvature solutions, and instead form shocks as discontinuous gradients rather than intensities. Though its results are appealing, LCIS smoothing is slow and its best published

results combine multiple LCIS images using as many as ten hand-selected parameters.

Nonlinear filter approaches are at least as old as Land's classic Retinex work, continued with Chiu and Shirley's [8] early work, and were recently advanced by an intriguing series of papers by Black *et al.* [5, 6], Tomasi and Manduchi [31] and Durand and Dorsey [10]. Black *et al.* [6] showed equivalence or strong parallels between iterative robust statistical methods and anisotropic diffusion [23]. Soon afterwards, Tomasi and Manduchi introduced the bilateral filter [31]. This simple, fast and elegant nonlinear filter performs good-quality edge-preserving smoothing in a single pass, and produces PDE-like results without an iterative solver or instability risks.

Unlike the iterative solvers, nonlinear filter methods compute each output pixel separately, as a position-dependent function of input pixels in a local neighborhood. Derivations by Barash [4], Elad [11] and confirmed by Durand and Dorsey [10] show that bilateral filtering is equivalent both to a single iteration of a discrete version of anisotropic diffusion and to several robust estimation methods. Durand and Dorsey [10] then demonstrated the value of the bilateral filter for contrast reduction, and used Fourifig:hdrer transform techniques to greatly accelerate it. However, the bilateral filter shares some of the drawbacks of anisotropic diffusion for contrast reduction.

In a different approach to contrast reduction, Fattal *et al.* [12] compressed the magnitude of the large image gradients that are responsible for its high contrasts, then iteratively solved a Poisson equation to find an image that best fits the compressed gradients in the least-squares sense. Their fast solver converges more quickly, requires fewer parameters, and avoids the often excessively 'busy' or noisy appearance of the LCIS method. All of these methods have guided our new work.

The trilateral filter is a substantially improved "detail-removing filter" for Figure 3 because it:

- better approximates scene illumination as a sharply-bounded, piecewise smooth signal with locally constant gradient,
- works in one pass, without an iterative PDE solver,
- forms sharp boundaries and corners much like shock forming PDEs,
- self-adjusts to the image, requiring one user-supplied parameter,
- extends easily to N-dimensional signals, both discrete and continuous-valued.

Polygonal mesh smoothing methods also apply nonlinear filtering techniques and iterative PDEs and were originally motivated by the problem of smoothing large irregular polygonal meshes of arbitrary topology [15, 17, 27] (see [30] for an excellent survey). Diffusion and curvature flow PDEs [3, 9] replaced initial Laplacian smoothing methods [27] and overcame its inherent mesh shrinkage problem. Tasdizen *et al.* [29] used a variational strategy to filter the surface normals instead of the point positions on the mesh. Their mesh smoothing method follows a 4-th order gradient-descent-based PDE.

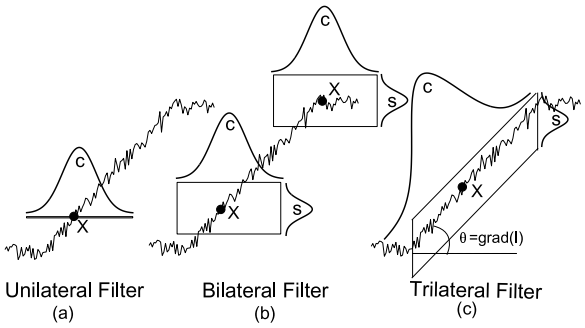

Figure 4: *Unilateral, Bilateral and Trilateral filter windows.*

Several recent papers used non-iterative nonlinear filters for denoising meshes, but their quality depends on how effectively the non-linear filter can emulate the behavior of complicated higher-order edge preserving PDEs. Peng *et al* . [21] and Alexa [1] have used Weiner filters for smoothing 3D meshes. Jones *et al.* [16] presented a two-pass approach that smoothes the face normals with a low-pass filter and then bilaterally smoothes the point positions in the mesh model using corrected normal information. Fleishman *et al.* [14] have also used the bilateral filter for smoothing the vertex locations of a 3D mesh model. We build on these methods to apply trilateral filters to meshes.

2. Filter Preliminaries

Linear and nonlinear filters make an output signal I_{out} by combining together neighboring parts of an input signal I_{in} in interesting and useful ways. The filters in this paper make weighted sums of neighboring values, but the weights and the neighborhoods may vary. Each filter described below is valid for N-dimensional inputs, but we will use 1-D and 2-D illustrations for clarity.

We begin with the linear "Finite Impulse Response"(FIR) or "unilateral" filter of Figure 4(a) to define our terms. The value of the filtered signal I_{out} at position $\mathbf{x} = (x, y, ...)$ is the integral of neighboring I_{in} values weighted by a filter kernel $c()$, or a weighted sum of nearby pixels for discrete input data. The offset vector ζ measures position in a local neighborhood or "domain" around $\mathbf{x}$, and the domain kernel function $c()$ provides a position-dependent scalar weight for each point's contribution to the output:

$$I_{out}(\mathbf{x}) = \int_{\infty}^{\infty} I_{in}(\mathbf{x}+\zeta)\, c(\zeta)\, d\zeta \qquad (1)$$

The domain kernel $c()$ may be any function, but we use the Gaussian function with variance σ_c for simplicity. Only the I_{in} points near $\mathbf{x}$ where ζ is small will receive a large weight,

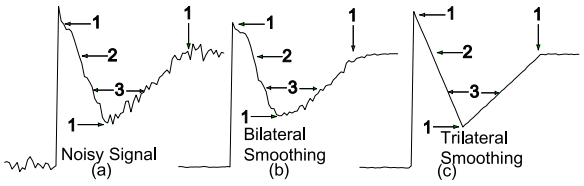

Figure 5: *Given a noisy piecewise linear signal (a), the bilateral filter blunts sharp corners (1b) and smoothes high gradient regions poorly (2b); the trilateral filter both sharpens corners (1c) and smoothes high gradient regions well (2c).*

and all points outside this "filter window," marked by a horizontal line in Figure 4(a), will have little effect on $I_{out}(\mathbf{x})$. Gaussian filter $c()$ removes details well, but also smoothes across the edges we wish to preserve.

Tomasi and Manduchi's bilateral filter [31] offers much better edge-preserving smoothing. As illustrated in Figure 4(b), it expands the filter window of domain $c()$ into a second dimension by multiplying with a "range" filter $s()$ that weights neighborhood values by their intensity difference from $I_{in}(\mathbf{x})$. If $s()$ is another Gaussian function with variance σ_s, then neighborhood I_{in} points with values nearly equal to $I_{in}(\mathbf{x})$ receive the highest $s()$ weight, but "outlier" points with greatly different values receive $s()$ weights near zero. Only the input points within the rectangular filter window shown in Figure 4(b) can strongly affect the output value $I_{out}(\mathbf{x})$:

$$I_{out}(\mathbf{x}) = \frac{1}{k(\mathbf{x})} \int_{-\infty}^{\infty} I_{in}(\mathbf{x}+\zeta)\underline{c(\zeta)s(I_{in}(\mathbf{x}+\zeta) - I_{in}(\mathbf{x}))}d\zeta \tag{2}$$

(Note underlined portions of Equations 2, 3 match). To ensure bilaterally filtered outputs are the average of similarly-valued nearby pixels, we normalize the neighborhood weights by $k(\mathbf{x})$:

$$k(\mathbf{x}) = \int_{-\infty}^{\infty} \underline{c(\zeta)s(I_{in}(\mathbf{x}+\zeta) - I_{in}(\mathbf{x}))}d\zeta. \tag{3}$$

Bilateral filters preserve most step-like edge features in I_{in} that are larger than the range variance σ_s because the filter window is not tall enough to include both the upper and lower portions of the step. However, the filter has serious drawbacks as a visual detail-removing filter because: (1) bilateral filters smooth across sharp changes in gradients, blunting or blurring ramp edges and valley- or ridge-like features (arrow 1, Fig. 5(b)), (2) high-gradient or high-curvature regions are poorly smoothed because most nearby I_{in} values are outliers that miss the filter window (arrow 2, Fig. 5(b)), and (3) wide bilateral filter windows may include disjoint domains on either side of adjacent high gradient regions as in Fig. 5 (arrow 3, Fig. 5(b)).

3. The Trilateral Filter

The new trilateral filter presented here combines two modified bilateral filters with a novel image-stack scheme for fast region-finding to avoid these problems. Its novel contributions are:

- *Tilting:* Its filter window is skewed or "tilted" by the bilaterally-smoothed image gradient vector G_θ in Figure 4(c), to track high-gradient regions (Section 3.1).
- *Adaptive Region-Growing:* The local neighborhood or "domain" automatically adapts to local image features to smooth the largest possible region with similar smoothed gradient values (Section 3.2).
- *One Parameter:* Though the trilateral filter uses 7 internal parameters (σ_c, $\sigma_{c\theta}$, σ_r, $\sigma_{r\theta}$, f_θ, R, β), all can be derived from a single user-supplied value $\sigma_{c\theta}$ (Section 3.3).

3.1. Tilting

As Figure 4(c) shows, the trilateral filter tilts its filter window by angle(s) θ, pivoting around the center point at $(\mathbf{x}, I(\mathbf{x}))$ to better fit the signal and widen the usable domain. Its tilting vector $G_\theta(\mathbf{x})$ should average together closely related neighborhood gradients, but should ignore nearby strongly dissimilar gradient outliers. Because bilateral filters are well suited to this task, we modify them to filter input image gradients:

$$G_\theta(\mathbf{x}) = \frac{1}{k_\theta(\mathbf{x})} \int_{-\infty}^{\infty} \nabla I_{in}(\mathbf{x}+\zeta)\underline{c(\zeta)s(\|\nabla I_{in}(\mathbf{x}+\zeta) - \nabla I_{in}(\mathbf{x})\|)}d\zeta \tag{4}$$

(underlined portions of Equations 4. 5 match)

$$k_\theta(\mathbf{x}) = \int_{-\infty}^{\infty} \underline{c(\zeta)s(\|\nabla I_{in}(\mathbf{x}+\zeta) - \nabla I_{in}(\mathbf{x})\|)}d\zeta. \tag{5}$$

We use forward differences instead of central differences to minimize the smoothing effect for approximating gradients in discrete images: $\nabla I_{in}(m,n) \approx (I_{in}(m+1,n) - I_{in}(m,n), I_{in}(m,n+1) - I_{in}(m,n))$.

Tilting the filter window in Figure 4(c) also confuses its definition, because the domain filter $c()$ and range filter $s()$ are no longer orthogonal. The solution is simple; rather than computing a range weight $s()$ for neighboring $I(\mathbf{x}+\zeta)$ by measuring its closeness to the center point value $I(\mathbf{x})$, instead we measure its closeness to a plane through $I(\mathbf{x})$, which acts as a "centerline" for the filter window of Figure 4c. Formally, this plane of intensity values $P(\mathbf{x},\zeta)$ defines the filter's input range as the first-order Taylor-Series approximation of neighborhood point values around $I_{in}(\mathbf{x})$. The plane orientation is set by the smoothed gradient vector G_θ instead of the ordinary gradient ∇I_{in}:

$$P(\mathbf{x},\zeta) = I_{in}(\mathbf{x}) + G_\theta \cdot \zeta \tag{6}$$

Note that $\mathbf{x}$, G_θ and ζ are all N-dimensional vectors. To compute trilateral filter output values $I_{out}(\mathbf{x})$, we subtract scalar value P from neighborhood I_{in} values to find a local detail signal $I_\Delta(\mathbf{x},\zeta)$. Instead of filtering the input signal as in

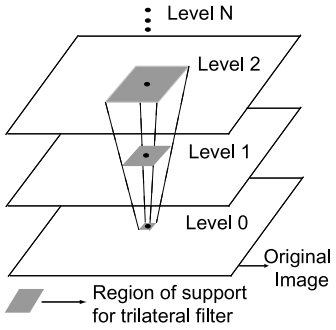

Figure 6: *We find neighborhood f_θ from a stack of min-max gradient images. Each pixel in level K holds the min and max values in a $(2^K+1)\times(2^K+1)$ size neighborhood of the level 0 image.*

Eqns. 2, 3 we apply the $c()$ and $s()$ weighting functions to $I_\Delta(\zeta)$ and add the result to $I_{in}(\mathbf{x})$ to form $I_{out}(\mathbf{x})$. The neighborhood used for each $\mathbf{x}$ is further restricted by the binary function $f_\theta(\mathbf{x},\zeta)=[0,1]$ explained in next section.

$$I_\Delta(\mathbf{x},\zeta) = I_{in}(\mathbf{x}+\zeta) - P(\mathbf{x},\zeta) \tag{7}$$

$$I_{out}(\mathbf{x}) = I_{in}(\mathbf{x}) + \frac{1}{k_\Delta(\mathbf{x})}\int_\infty^\infty I_\Delta(\mathbf{x},\zeta)\underline{c(\zeta)s(I_\Delta(\mathbf{x},\zeta))f_\theta(\mathbf{x},\zeta)}d\zeta \tag{8}$$

The trilateral filter also normalizes local weights by $k_\Delta(x)$ (underlined portions of Equations 7, 8 match):

$$k_\Delta(\mathbf{x}) = \int_\infty^\infty \underline{c(\zeta)s(I_\Delta(\mathbf{x},\zeta))f_\theta(\mathbf{x},\zeta)}d\zeta \tag{9}$$

3.2. Automatic f_θ: Adaptive Neighborhood

Tilting greatly improves smoothing abilities of the trilateral filter in high gradient regions, but also ensures that the filter window can extend beyond local boundaries into regions of dissimilar gradients. Unless we exclude these regions from the filter window, the trilateral filter will blunt or blur sharp ridges and corner-like features where the bilaterally smoothed gradient G_θ changes abruptly (e.g. arrow 1 in Figure 5b). Tilting is not enough: we need the "edge-limited smoothing" effects offered by the shocks (zero conductance boundaries) that form in anisotropic diffusion [23] or LCIS [33]. Fortunately, the G_θ signal itself contains the solution. The smooth, approximately piecewise-constant magnitude $\|G_\theta\|$ forms step-like features at $\mathbf{x}$ locations where I_{in} has ridge and corner-like transitions. We apply a threshold R to these features to form a binary signal $f_\theta(\mathbf{x},\zeta)$ used in Equations 8, 9 that limits the smoothed neighborhood to connected regions $\mathbf{x}$ that share similar $\|G_\theta\|$ vectors. "Similar" values are decided by scalar threshold parameter R (see Section 3.3):

$$f_\theta(\mathbf{x},\zeta) = \begin{cases} 1 & \text{if } \|G_\theta(\mathbf{x}+\zeta) - G_\theta(\mathbf{x})\| < R \\ 0 & \textit{otherwise.} \end{cases} \tag{10}$$

Searching the neighborhood around $\mathbf{x}$ for connected regions of nonzero f_θ is expensive. Instead, we approximate these regions by their largest inscribed square, and we can find this square by a simple lookup operation in a min-max image stack. If input data I_{in} is an image holding $N\times M$ pixels, the stack is a $log_2(N)$ set of $N\times M$ images, each one called a "level" and numbered upwards from zero, as shown in Figure 6. Unlike an image pyramid, each image in a stack is the same $N\times M$ size, but effective filter sizes still double with each successive level. Level 0 pixels hold the original $G_\theta(\mathbf{x})$ vector elements, the level 1 pixel at (m,n) holds min and max values for each $G_\theta(\mathbf{x})$ element in the surrounding 3×3 pixels found in level 0 at $(m+[0,\pm 2^0], n+[0,\pm 2^0])$, level 2 pixels hold min and max values for the surrounding 3×3 pixels found in level 1 at $(m+[0,\pm 2^1], n+[0,\pm 2^1])$. Formally, each pixel (m,n) in any nonzero level K holds min and max values for the 3×3 surrounding pixels found in level $(K-1)$ at $(m+[0,\pm 2^{k-1}], n+[0,\pm 2^{k-1}])$. Because these 3×3 neighborhoods overlap, level K pixels each hold min and max values for (2^K+1) by (2^K+1) surrounding pixels of level 0.

To find the connected region of nonzero f_θ around $\mathbf{x}$, we traverse the image stack at pixel $\mathbf{x}$ to find the highest level whose min/max values are within $G_\theta(\mathbf{x})\pm R$. Though we found it unnecessary, it is possible to iteratively expand this inscribed-square solution to find the complete connected f_θ region. Starting from the level K pixel (m,n), find adjacent inscribed squares by testing nearby pixels in adjacent levels. Iteratively test level $K\pm 1$ pixels at $(m+[0,\pm 2^{K\pm 1}], n+[0,\pm 2^{K\pm 1}])$ to enlarge the connected set of qualified stack pixels that describe the region. Using more than one min-max image stack pixel may prove useful to some applications that need better f_θ approximations near important but noisy diagonal edges.

3.3. Self-Adjusting Parameters

Avoiding hand-tuned parameters improves the usefulness and generality of the trilateral filter, and it requires one user-specified parameter: ($\sigma_{c\theta}$), the neighborhood size used for bilateral gradient smoothing. More intuitively, $\sigma_{c\theta}$ sets the typical size of separately-smoothed regions in the output image. The trilateral filter's single parameter offers improvement over both the bilateral filter [31] with 3 parameters (the domain variance, σ_c, the range variance σ_s and the width of the filter kernel f) and LCIS [33] with about 3 sets of 3 parameters each (timestep, edginess factor g and number of iterations). Though the trilateral filter has 7 internal parameters, values for all but one are computed automatically.

The trilateral filter uses two bilateral stages and a min-max stack. The parameter-setting procedure begins with the user-supplied $\sigma_{c\theta}$ value; large $\sigma_{c\theta}$ expands the spatial extent, but may blur or blunt boundaries where only the gradient changes. First, we use $\sigma_{c\theta}$ as the radius of a circular neighborhood around $\mathbf{x}$ in the input image. We find the aver-

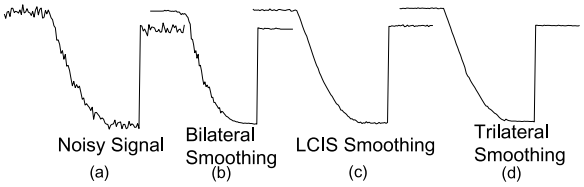

Figure 7: *Bilateral smoothing can blunt sharp corners and smoothes high gradient regions poorly. Trilateral filter, like LCIS, drives the final signal towards a piecewise linear approximation.*

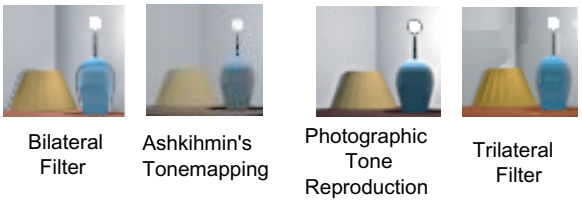

Figure 8: *Small adjacent high gradient image regions near the lamp-base top, wall corner and bulb cause difficulties for many previous methods. Image excerpt from a larger bulb scene, courtesy of Peter Shirley, University of Utah.*

age gradient G_{avg} in this neighborhood for each $\mathbf{x}$, and then use the min and max G_{avg} as an estimate of overall gradient variability. This variability defines outliers for gradients that will be rejected by $\sigma_{s\theta}$:

$$\sigma_{s\theta} = \beta(\|\max(G_{avg}(\mathbf{x})) - \min(G_{avg}(\mathbf{x}))\|) \qquad (11)$$

Large $\sigma_{s\theta}$ improves noise reduction, but also reduces outlier rejection, and may blur weaker boundaries of slight intensity changes. Unfortunately, β is a small fraction we set empirically to 0.15; values between 0.1 and 0.2 always worked best. Armed with $\sigma_{c\theta}$ and $\sigma_{s\theta}$, we then compute the min-max stack of Section 3.2. We set the globally-applied region-finding threshold by $R = \sigma_{s\theta}$ to ensure region size f_θ does not include gradient outliers excluded from the bilateral filtering.

Finally, we compute the trilateral filter output from Equations 7,8, and 9. Domain filtering for I_{in} uses the same neighborhood size applied earlier for gradient filtering: $\sigma_c = \sigma_{c\theta}$. The range filtering variance is more interesting, because the trilateral filter smoothes detail I_Δ measured from the plane P of Equation 6. The amplitude of the detail signal is closely related to the variance of the gradients or the difference between the smoothed gradient G_θ and the actual gradients. Thus we can re-use our definition for gradient outliers: $\sigma_s = \sigma_{s\theta}$. These simple rules have proven surprisingly robust for a wide variety of signal classes, including images and 3D geometric meshes.

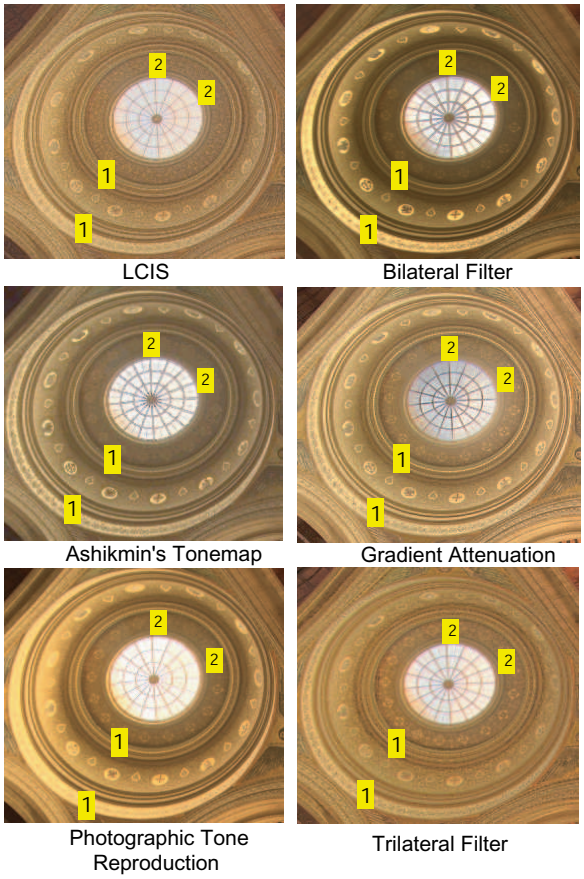

Figure 9: *Despite high contrasts and strongly varying gradients, the trilateral filter preserves details that escape many previous methods. Note the ring-like specular highlight (1) that often escapes contrast compression. Only the trilateral filter and gradient attenuation [12] methods capture the subtle gold medallions (2) and radial lines near the skylight. Image excerpt from larger Stanford Church scene, courtesy of Paul Debevec, University of Southern California.*

4. Results

In this section, we apply the trilateral filter to the tasks of displaying high contrast images and de-noising 3D mesh models.

4.1. HDR Tone Mapping or Contrast Reduction

The trilateral filter offers several notable improvements when used for high dynamic range (HDR) tone mapping or contrast reduction. We collected several HDR source images from previous tone-mapping papers and applied the trilateral filter of Equations 7, 8, 9 in the "base/detail" method shown in Figure 3. In side-by-side comparisons with five other recently published methods [2, 10, 12, 24, 33] the differences are instructive.

Figure 10: *Nonuniform gradient compression [12] can sometimes lead to brightness anomalies. Image courtesy of Shree Nayar, Columbia University.*

Figure 11: *Trilateral filter automatically selects all but one parameter; adjusting for good results is relatively easy. Images courtesy of Dani Lischinski, Hebrew University, Israel.*

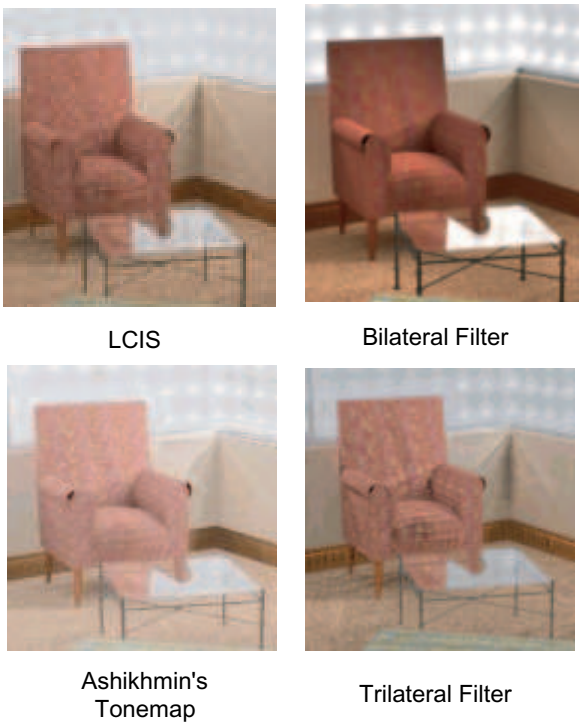

Figure 12: *Halo artifacts from bilateral filtering [10] at the top of the sofa and chair from are absent in the other methods. Image courtesy of Simon Crone, Perth, Australia.*

Figure 1, 8, 10 and others show that the trilateral filter is particularly good at edge-preserving smoothing in narrow ramp-like high gradient regions of an image, such as the shading at the top of the lamp base. Here, as in Figure 5(arrow 3), the bilateral filter can span different high gradient regions and cause strange, strip-like bipolar halos. Though Ashikhmin's method [2] is more successful, it blurs the lamp slightly and makes the lampshade and wall boundaries indistinct. Conversely, photographic tone mapping by Reinhard *et al.* [24] keeps the image sharp, but surrounds the bare lightbulb with a thin black halo.

Like LCIS [33], the trilateral filter smoothes towards a piecewise constant gradient or low curvature result, and in most Figures (e.g. 9, 10, 13) trilateral results more closely resemble LCIS than any other. But LCIS works by iterative smoothing, and require many hand-selected parameters, and poor choices can lead users to washed-out, overly busy results as in Figure 11, but the single-parameter trilateral filter easily provides more pleasing results.

As Figure 5(arrow 1) shows, tilting and adaptive neighborhoods help trilateral filters preserve large, sharp gradient changes. Smoothing across these changes causes a dark halo at the chair top in the bilateral results in Figure 12. Adaptive neighborhoods help the trilateral filter smooth well even near very high-contrast features, enabling preservation of very subtle medallions and radial lines in the decorative rings nearest the skylight border in Figure 9 at area 2. Even the gradient attenuation method [12] results loses some details here. As in Figure 5, bilateral filtering [10] blunts the sharp specular highlights in an outer ring, permitting uncompressed brilliance in the result ("1" in Figure 9). LCIS [33] over-emphasizes some details near the skylight and somehow lost to the subtle golden medallions revealed by trilateral filter (at "2" in Figure 9).

The trilateral filter also avoids blooming effects that enlarge, blur or brighten high gradient neighborhoods, such as the inner ring of the skylight for Figure 9. Figure 10 seems to show some blooming-like brightness anomalies due to nonuniform compression of image gradients [12] that are not reproduced by the trilateral filter. Figure 13 demonstrates that blooming (at arrow) for extremely high contrast specular reflections can be difficult to avoid in the bilateral filter [10], but both the trilateral filter and gradient attenuation method [12] nearly match the blooming suppression of the other three methods.

Table 4.1 shows the computation time and the half of the

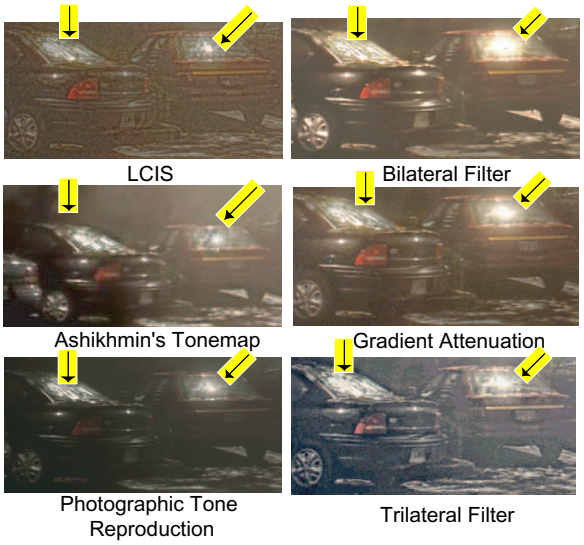

Figure 13: *Compared to bilateral filter [10], trilateral filtering limits blooming of sharp specular highlights and its performance is similar to the gradient attenuation [12] method.*

Figure# : Size	Bilateral		Trilateral	
	f_θ	Time(s)	f_θ	Time(s)
8: 400×300	10	13.13	6.67	21.2
13: 1130×750	10	110.1	5.4	179.3
9: 512×768	10	57.2	4.95	89.1
11: 1024×768	10	100.5	5.9	160.1
12: 750×485	10	51.4	6.6	90.2

Table 1: *Running time and average size of adaptive neighborhood (kernel radius f_θ in pixels) for trilateral filter.*

adaptive filter kernel f_θ for the bilateral and the trilateral filter. Theoretically, the trilateral filter should take roughly twice the time of bilateral filter as it is a two-step bilateral filtering procedure. In practice, the running time for the trilateral filter is a little less, due to variable neighborhood size f_θ. In Table 4.1, the filter window half-width f_θ is constant for the bilateral filter for all the images. For the trilateral filter f_θ varies with the scene details and the average filter window half-width is often smaller than the bilateral filter value. All running times measure non-optimized code for both the bilateral and the trilateral filter. The Fourier transform and sub-sampling based acceleration techniques devised by Durand and Dorsey [10] should greatly reduce the running times for both filters.

4.2. Mesh Smoothing

The trilateral filter can also perform 3D mesh smoothing. Though several mesh smoothing approaches are possible, we chose a two-step process: first, trilateral normal filtering (Section 4.2.1) computes the new vertex normals $\mathbf{N}_{Vout}$ and defines a "filter plane" $P_V(\mathbf{X}_V,\zeta)$ for each. Then trilateral vertex filtering (Section 4.2.2) smoothes together distances from the filter plane to mesh faces in its neighborhood, and we use this distance to find a new vertex position $\mathbf{X}_{Vout}$ along the new mesh normal direction.

Note that our method for smoothing each mesh vertex V requires both its position $\mathbf{X}_V=(X,Y,Z)$ and surface normal vector $\mathbf{N}_V$. If normals are unknown, then $\mathbf{N}_V$ is typically computed as the area weighted average of normals for incident faces of V [14].

4.2.1. Trilateral Normal Filtering

To find new vertex normals, begin by bilaterally filtering the given vertex normals $\mathbf{N}_V$ with Equations 4 and 5. Simply substitute normal vectors $\mathbf{N}_V$ for gradient vectors ∇I_{in}, use the domain filter $c()$ to weight the contribution of each nearby vertex's normal according to its 3D distance from vertex V, and let the range filter $s()$ assign weights that will reject outlier directions for normals. The resulting bilaterally smoothed normals $\mathbf{N}_{\theta V}$ allow us to find a connected neighborhood of nearby mesh faces with similar normals.

For trilateral normal filtering, we refer to each mesh face near vertex V by the name ζ_F, and its face normal and face center point is $\mathbf{X}_{\zeta F}$ and $\mathbf{N}_{\zeta F}$ respectively. As before, function $f_\theta()$ defines the adaptive neighborhood around vertex V, and its limited extent ensures that the trilateral filter window will not cross sharp corners of the mesh during filtering. Function $f_\theta(V,\zeta_F)$ is 1 for all connected neighborhood faces around V that share normal vectors similar to $\mathbf{N}_{\theta V}$, and is otherwise zero. Breadth-first search implemented as a region growing algorithm finds this connected neighborhood. The traversal starts at vertex V and terminates when all surrounding face normals $\mathbf{N}_{\zeta F}$ differ significantly from the vertex normal $\mathbf{N}_{\theta V}$:

$$f_\theta(V,\zeta_F)=\begin{cases}1 & \text{if } \|\mathbf{N}_{\theta V}\cdot\mathbf{N}_{\zeta F}\|<R\\ 0 & otherwise.\end{cases} \tag{12}$$

The bilateral filtered normal $\mathbf{N}_{\theta V}$ also sets the filter plane for each vertex V. Analogous to the 'centerline' in Section 3.1, planeeq:trilat2 $P_V(\mathbf{X}_V,\zeta)$ passes through vertex V and is perpendicular to the bilaterally smoothed normal $\mathbf{N}_{\theta V}$. Unlike the plane P in Equation 6 which provides a range value for a given location $\mathbf{x}$, the plane $P_V(\mathbf{X}_V\zeta)$ is defined only in the 3D domain, and the detail signal is set only by the distance to that plane.

$$P_V(\mathbf{X}_V,\zeta)satisfies(\zeta-\mathbf{X}_V)\cdot\mathbf{N}_{\theta V}=0 \tag{13}$$

We compute the trilaterally filtered normal $\mathbf{N}_{Vout}$ for vertex V from the filter plane P_V and the normals of neighboring

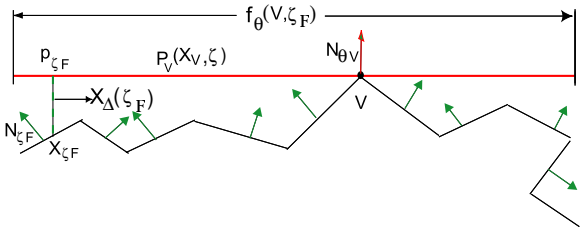

Figure 14: *The bilaterally filtered normal $\mathbf{N}_{\theta V}$ defines a center plane $P_V(\mathbf{X}_V,\zeta)$ through each vertex V. The adaptive region f_θ selects nearby faces with similar normals. Distance from each face center $\mathbf{X}_{\zeta F}$ to the plane defines the detailed distance signal $X_\Delta(\zeta_F)$.*

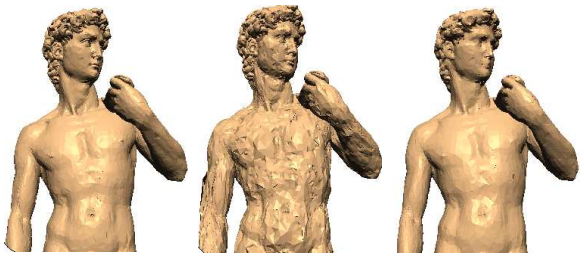

Figure 15: *In a single pass, a trilateral filter can remove most visible corruptions caused by additive Gaussian noise in both vertex positions and normals. Some small high curvature creases were lost due to smoothing in the hair, eyelids and lips.*

faces selected by F_θ. The filter smoothes a "normal detail signal" $\mathbf{N}_\Delta(\zeta_F)$ made from differences with neighborhood face normals $\mathbf{N}_{\zeta F}$ around the vertex V:

$$\mathbf{N}_\Delta(\zeta_F) = \mathbf{N}_{\theta V} - \mathbf{N}_{\zeta F} \tag{14}$$

The domain filter $c_N(\zeta_F)$ is a Gaussian weighting function that falls towards zero as the 3D distance $\|X_V - X_{\zeta F}\|$ from vertex to face center increases. The range filter $s_N(\mathbf{N}_\Delta(\zeta_F))$ gives low weights to outlier face normal directions $\mathbf{N}_{\zeta F}$ that are drastically different from $\mathbf{N}_{\theta V}$. Echoing Equation 8, the trilaterally filtered normal $\mathbf{N}_{Vout}$ is then:

$$\mathbf{N}_{Vout} = \mathbf{N}_{\theta V} + \frac{1}{k_N(\zeta_F)} \sum_{F\in\zeta_F} \mathbf{N}_\Delta(\zeta_F)\underline{c_N(\zeta_F)s_N(\mathbf{N}_\Delta(\zeta_F))f_\theta(V,\zeta_F)}, \tag{15}$$

where the weighting coefficients of the trilateral filter are normalized by $k_N(\zeta_F)$:

$$k_N(\zeta_F) = \sum_{F\in\zeta_F} \underline{c_N(\zeta_F)s_N(\mathbf{N}_\Delta(\zeta_F))f_\theta(V,\zeta_F)} \tag{16}$$

4.2.2. Vertex Filtering

Next, for each mesh vertex V we find a new vertex position $\mathbf{X}_{Vout}$ by additional trilateral filtering using results from the previous section. Vertex filtering re-uses the same adaptive neighborhood f_θ and the same filter plane P_V, but computes a scalar distance for each vertex. Displacing the vertex by this distance in the new normal direction $\mathbf{N}_{Vout}$ produces the new vertex position $\mathbf{X}_{Vout}$. The vertex filter finds a domain- and range-weighted average of the "distance detail signal" $X_\Delta(\zeta_F)$ that measures the distance from the filter plane $P_V(\mathbf{X}_V,\zeta)$ to each face center $\mathbf{X}_{\zeta F}$ in the f_θ neighborhood around. Formally, define the point $\mathbf{p}_{\zeta F}$ as the projection of the face center point $\mathbf{X}_{\zeta F}$ onto the plane $P_V(\mathbf{X}_V,\zeta)$, as shown in Figure 14, and then define the distance detail signal $X_\Delta(\zeta_F)$ as the 3D distance between $\mathbf{p}_{\zeta F}$ and the face center point $\mathbf{X}_{\zeta F}$:

$$X_\Delta(\zeta_F) = \| \mathbf{p}_{\zeta F} - \mathbf{X}_{\zeta F}\| \tag{17}$$

To smooth X_Δ properly, the domain filter $c_V(\mathbf{p}_{\zeta F})$ is a Gaussian weighting function that falls towards zero as the 2D distance $\|\mathbf{X}_V - \mathbf{p}_{\zeta F}\|$ from vertex to point p increases. The range filter $s_V(X_\Delta(\zeta_F))$ is a Gaussian weighting that rejects outlier face centers $\mathbf{X}_{\zeta F}$ that are too far away from the plane $P_V(\mathbf{X}_V,\zeta)$. We use the same $f_\theta(V,\zeta_F)$ as before in Equations 8, 9. Each new output vertex position is:

$$\mathbf{X}_{Vout} = \mathbf{X}_V + \frac{\mathbf{N}_{Vout}}{k_V(\zeta_F)} \sum_{F\in\zeta_F} \mathbf{X}_\Delta(\zeta_F)\underline{c_V(\mathbf{p}_{\zeta F})s_V(X_\Delta(\zeta_F))f_\theta(V,\zeta_F)} \tag{18}$$

The weighting term is normalized by

$$k_V(\zeta_F) = \sum_{F\in\zeta_F} \underline{c_V(\mathbf{p}_{\zeta F})s_V(X_\Delta(\zeta_F))f_\theta(V,\zeta_F)} \tag{19}$$

4.2.3. Mesh Smoothing Results

Figure 2 shows an artificially corrupted dragon face model before and after trilateral smoothing. Trilateral filter retains most of the sharp curvatures in the face of the dragon. Figure 15 shows the effect of trilateral smoothing on the noisy David input model. Leaving aside a little blurring in the eyelid and hair of the input David model, our filter preserves sharp features throughout the model. Figure 16 compares the results for mesh denoising using trilateral filter with two recent mesh smoothing algorithms [16, 14]. All the three methods efficiently smoothes the noisy input mesh, though the result of Fleishman *et al.*'s algorithm [14] is comparable in quality to the trilateral filter perhaps because both the algorithms filter the tangent plane distance for neighborhood points. Figure 17 shows the results of smoothing a different input model for the trilateral filter and the modified bilateral filter proposed by Jones *et al.* [16]. The performance of both the algorithms are roughly similar, but some minute differences are visible around the ear and mouth outlines.

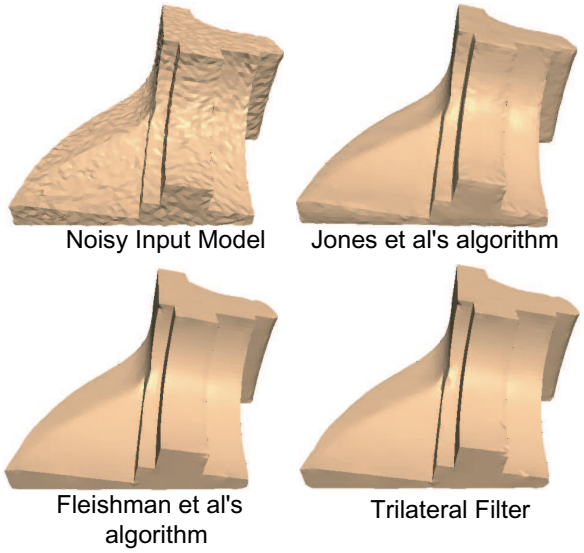

Figure 16: *All the three algorithms are effective for denoising the noisy Fandisk model, but the trilateral filter result is closely matched to Fleishman* et al.*'s* [14] *result.*

Figure 17: *Jones* et al.*'s* [16] *smoothing algorithm and trilateral filter produce similar results except for small differences at the edges of the dog's ears and lips.*

5. Conclusions and Future Work

The trilateral filter offers new edge-preserving detail-remover that smoothes input towards a piecewise constant gradient approximation. The filter requires only one user-specified parameter and is applicable to N-dimensional data. We demonstrated its usefulness for different applications like high contrast image display and mesh smoothing. The filter is also "embarrassingly parallel" and may prove suitable for fast hardware implementation.

Trilateral filter's ability to separate details from the noisy original image and to predict gradient discontinuities in spatial domain with sub-pixel accuracy might prove useful in image and photo-editing operations [20, 22]. The filter might also benefit from a more pricipled justification for the constant β in Eqn. 11. The trilateral filter extended to the spatio-temporal domain can also predict occlusions in temporal domain and this feature has potential for various video-based-rendering applications [13, 34].

Acknowledgments:

The authors thank Michael Ashikhmin, Simon Crone, Fredo Durand, Paul Debevec, Rannan Fattal, Dani Lischinski, Erik Reinhard and Greg Ward for providing original high dynamic range images, results, and permission to use them. We are also grateful to Mathieu Desbrun, Shachar Fleishman and Ray Jones for providing mesh and results data for comparison with their current mesh smoothing algorithms.

References

1. M. Alexa, "Weiner filtering of meshes," in *Proc. Shape Modeling International (SMI)*, pp. 51-57, 2002. 3
2. M. Ashikhmin, "A tone mapping algorithm for high contrast images," in *P. Debevec and S. Gibson Eds., Proc. 13th Eurographics Workshop on Rendering (EGRW)*, pp. 145-156, 2002. 1, 2, 6, 7
3. C. Bajaj and G. Xu, "Anisotropic diffusion of surfaces and functions on surfaces," *ACM Trans. Computer Graphics*, vol. 22(1), pp. 4-32, 2003. 3
4. D. Barash, "A fundamental relationship between bilateral filtering, adaptive smoothing and nonlinear diffusion equation," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 24(6), pp. 844-850, 2002. 3
5. M. Black, and A. Rangarajan, "On the unification of line processes, outlier rejection, and robust statistics with applications in early vision," *International Journal of Computer Vision*, vol. 19(1), pp. 57-92, 1996. 1, 3
6. M. J. Black, G. Sapiro, D. Marimont, and D. Heeger, "Robust anisotropic diffusion," *IEEE Transactions on Image Processing*, vol. 7(3), pp. 421-432, 1998. 1, 3
7. P. Debevec, and J. Malik, "Recovering high dynamic range radiance maps from photographs," in *Proc. SIGGRAPH 97, ACM SIGGRAPH / Addison Wesley Longman, Computer Graphics Proceedings, Annual Conference Series*, pp. 369-378, 1997. 1
8. K. Delvin, A. Chalmers, A. Wilkie and W. Purgathofer, "Tone reproduction and physically based spectral rendering," *EUROGRAPHICS 2002: State of the Art Report (STAR)*, 2002. 1, 2, 3
9. M. Desbrun, M. Meyer, P. Schroeder and A. H. Barr, "Implicit fairing of irregular meshes using diffusion and curvature flow," *Proc. of ACM SIGGRAPH'99 Conference Proceedings*, pp. 317-324, 1999. 3
10. F. Durand, and J. Dorsey, "Fast bilateral filtering for the display of high-dynamic range images," *ACM Transactions on Graphics, special issue on Proc. of ACM SIGGRAPH 2002, San Antonio, Texas*, vol. 21(3), pp. 249-256, 2002. 1, 2, 3, 6, 7, 8

11. M. Elad, "On the bilateral filter and ways to improve it," *IEEE Transaction Image Processing*, vol. 11(10), pp. 1141-1151, 2002. 1, 3

12. R. Fattal, D. Lischinski, and M. Werman, "Gradient domain high dynamic range compression," *ACM Transactions on Graphics, special issue on Proc. of ACM SIGGRAPH 2002, San Antonio, Texas*, vol. 21(3), pp. 257-266, 2002. 3, 6, 7, 8

13. A. Finkelstein, C. E. Jacobs and D. H. Salesin, "Multiresolution video," *Proc. of ACM SIGGRAPH 1996, ACM SIGGRAPH, Computer Graphics Proceedings, Annual Conference Series*, pp. 281-290 , 1996. 10

14. S. Fleishman, I. Drori and D. Cohen-Or, "Bilateral mesh denoising," *ACM Transactions on Graphics, special issue on Proc. of ACM SIGGRAPH 2003, San Diego, California*, 2003 (to appear). 3, 8, 9, 10

15. I. Guskov, W. Sweldens and P. Schroeder, "Multiresolution signal processing for meshes," in *Proc. SIGGRAPH 99, ACM SIGGRAPH, Los Angeles, California, Computer Graphics Proceedings, Annual Conference Series*, pp. 325-334, 1999. 3

16. T. R. Jones, F. Durand and M. Desbrun, "Non-iterative feature preserving mesh smoothing," *ACM Transactions on Graphics, special issue on Proc. of ACM SIGGRAPH 2003, San Diego, California*, 2003 (to appear). 3, 9, 10

17. L. Kobbelt, S. Campagna, J. Vorsatz and H. P. Siedel, "Interactive multi-resolution modeling on arbitrary meshes," in *Proc. SIGGRAPH 98, ACM SIGGRAPH, Computer Graphics Proceedings, Annual Conference Series*, pp. 105-114, 1998. 3

18. A. McNamara, "Visual perception in realistic image synthesis," *EUROGRAPHICS 2000: State of the Art Report (STAR)*, Interlaken, Switzerland, 2000. 1, 2

19. S. K. Nayar and T. Mitsunaga, "High dynamic range imaging: spatially varying pixel exposures," in *Proc. Computer Vision and Pattern Recognition (CVPR)*, pp. 473-479, 2000. 1

20. B. M. Oh, M. Chen, J. Dorsey and F. Durand, "Image based modeling and photo editing," in *Proc. SIGGRAPH 2001, ACM SIGGRAPH, Los Angeles, California, Computer Graphics Proceedings, Annual Conference Series*, pp. 433-442, 2001. 10

21. J. Peng, V. Strela, D. Zorin, "A simple algorithm for surface denoising," *Proceedings of IEEE Visualization* 2001. 3

22. P. Perez, M. Gangnet and A. Blake, "Poisson image editing," *ACM Transactions on Graphics, special issue on Proc. of ACM SIGGRAPH 2003, San Diego, California*, 2003 (to appear). 10

23. P. Perona, and J. Malik, "Scale space and edge detection using anisotropic diffusion," *IEEE Transaction Pattern Analysis and Machine Intelligence*, vol. 12(7), pp. 629-639, 1990. 1, 2, 3, 5

24. Reinhard, E., Stark, M., Shirley, P. and Ferwada, J., "Photographic tone reproduction for digital images," *ACM Transactions on Graphics, special issue on Proc. of ACM SIGGRAPH 2002, San Antonio, Texas*, vol. 21(3), pp. 267-276, 2002. 1, 2, 6, 7

25. G. Sapiro, "Geometric partial differential equations and image analysis,"*Cambridge University Press*, 2001. 1

26. C. Schlick, "Quantization techniques for visualization of high dynamic range pictures," in *G. Sakas, et al. eds. Photorealistic Rendering Techniques, Proc. 5th Eurographics Rendering Workshop*, pp. 7-20, 1995. 2

27. G. Taubin, "A signal processing approach to fair surface design," in *Proc. SIGGRAPH 95, ACM SIGGRAPH, Computer Graphics Procedings, Annual Conference Series*, pp. 351-358, 1995. 3

28. T. G. Stockham, "Image processing in the context of a visual model," *Proceedings of the IEEE*, vol. 60, no. 7, pp. 828-842, 1972. 2

29. T. Tasdizen, R. Whitaker, P. Burchard and S. Osher, "Geometric surface processing via normal maps," *ACM Trans. Computer Graphics*, Sepetmber 2003 (to appear). 3

30. G. Taubin, "Geometric signal processing on polygonal meshes," *EUROGRAPHICS 2000: State of the Art Report (STAR)*, Interlaken, Switzerland, 2000. 3

31. C. Tomasi, and R. Manduchi, "Bilateral filtering of gray and colored images," *Proc. IEEE Intl. Conference on Computer Vision*, pp. 836-846, 1998. 1, 3, 4, 5

32. J. Tumblin, and H. Rushmeier, "Tone reproduction for realistic images," *IEEE Computer Graphics and Applications*, vol. 13(6), pp. 42-48, 1993. 1

33. J. Tumblin and G. Turk, "LCIS: A boundary hierarchy for detail-preserving contrast reduction," in *Proc. SIGGRAPH 99, ACM SIGGRAPH, Los Angeles, California, Computer Graphics Proceedings, Annual Conference Series*, pp. 83-90, 1999. 1, 2, 5, 6, 7

34. S. Vedula, S. Baker and T. Kanade, "Spatio-temporal view interpolation ", *Proc. 13th ACM Eurographics Workshop on Rendering*, pp. 65-76, 2002. 10

35. R. Whittaker and S. Pizer, "A multi-scale approach to nonuniform diffusion," *CVGIP: Image Understanding*, vol. 57(1), pp. 99-110, 1993. 1, 2

Figure 18: *Examples of high dynamic range radiance map compression and mesh smoothing using trilateral filter. (Top Row, from left): Stanford Memorial Church, courtesy of Paul Debevec, Univ. Southern California. Tree on a Foggy Night, Washington DC Cathedral, courtesy of Max Lyons. (Middle Row): Synagogue, courtesy of Dani Lischinski, Hebrew University, Israel, Burswood Hotel Suite Refurbishment, ©1995 Simon Crone. (Bottom Row): Noisy Venus model and its smoothed version using trilateral filter.*