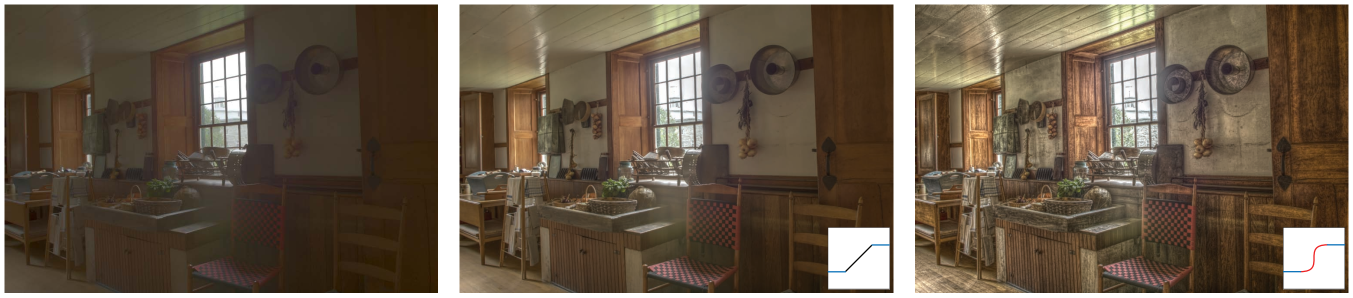
Local Laplacian Filters: Edge-aware Image Processing with a Laplacian Pyramid

Sylvain Paris
Adobe Systems, Inc.

Samuel W. Hasinoff
Toyota Technological Institute at Chicago and MIT CSAIL

Jan Kautz
University College London



(a) input HDR image tone-mapped with a simple gamma curve (details are compressed) (b) our pyramid-based tone mapping, set to preserve details without increasing them (c) our pyramid-based tone mapping, set to strongly enhance the contrast of details

Figure 1: *We demonstrate edge-aware image filters based on the direct manipulation of Laplacian pyramids. Our approach produces high-quality results, without degrading edges or introducing halos, even at extreme settings. Our approach builds upon standard image pyramids and enables a broad range of effects via simple point-wise nonlinearities (shown in corners). For an example image (a), we show results of tone mapping using our method, creating a natural rendition (b) and a more exaggerated look that enhances details as well (c). Laplacian pyramids have previously been considered unsuitable for such tasks, but our approach shows otherwise.*

Abstract

The Laplacian pyramid is ubiquitous for decomposing images into multiple scales and is widely used for image analysis. However, because it is constructed with spatially invariant Gaussian kernels, the Laplacian pyramid is widely believed as being unable to represent edges well and as being ill-suited for edge-aware operations such as edge-preserving smoothing and tone mapping. To tackle these tasks, a wealth of alternative techniques and representations have been proposed, e.g., anisotropic diffusion, neighborhood filtering, and specialized wavelet bases. While these methods have demonstrated successful results, they come at the price of additional complexity, often accompanied by higher computational cost or the need to post-process the generated results. In this paper, we show state-of-the-art edge-aware processing using standard Laplacian pyramids. We characterize edges with a simple threshold on pixel values that allows us to differentiate large-scale edges from small-scale details. Building upon this result, we propose a set of image filters to achieve edge-preserving smoothing, detail enhancement, tone mapping, and inverse tone mapping. The advantage of our approach is its simplicity and flexibility, relying only on simple point-wise nonlinearities and small Gaussian convolutions; no optimization or post-processing is required. As we demonstrate, our method produces consistently high-quality results, without degrading edges or introducing halos.

Keywords: image pyramids, edge-aware image processing

1 Introduction

Laplacian pyramids have been used to analyze images at multiple scales for a broad range of applications such as compression [Burt and Adelson 1983], texture synthesis [Heeger and Bergen 1995], and harmonization [Sunkavalli et al. 2010]. However, these pyramids are commonly regarded as a poor choice for applications in which image edges play an important role, e.g., edge-preserving smoothing or tone mapping. The isotropic, spatially invariant, smooth Gaussian kernels on which the pyramids are built are considered almost antithetical to edge discontinuities, which are precisely located and anisotropic by nature. Further, the decimation of the levels, i.e., the successive reduction by factor 2 of the resolution, is often criticized for introducing aliasing artifacts, leading some researchers, e.g., Li et al. [2005], to recommend its omission. These arguments are often cited as a motivation for more sophisticated schemes such as anisotropic diffusion [Perona and Malik 1990; Aubert and Kornprobst 2002], neighborhood filters [Tomasi and Manduchi 1998; Kass and Solomon 2010], edge-preserving optimization [Bhat et al. 2010; Farbman et al. 2008], and edge-aware wavelets [Fattal 2009].

While Laplacian pyramids can be implemented using simple image resizing routines, other methods rely on more sophisticated techniques. For instance, the bilateral filter relies on a spatially varying kernel [Tomasi and Manduchi 1998], optimization-based methods, such as [Fattal et al. 2002; Farbman et al. 2008; Subr et al. 2009; Bhat et al. 2010], minimize a spatially inhomogeneous energy, and other approaches build dedicated basis functions for each new image, e.g., [Szeliski 2006; Fattal 2009; Fattal et al. 2009]. This additional level of sophistication is also often associated with practical shortcomings. The parameters of anisotropic diffusion are difficult to set because of the iterative nature of the process, neighborhood filters tend to over-sharpen edges [Buades et al. 2006], and methods based on optimization do not scale well due to the algorithmic complexity of the solvers. While some of these shortcomings can be alleviated in post-processing, e.g., bilateral filtered edges can be smoothed [Durand and Dorsey 2002; Bae et al. 2006; Kass and Solomon 2010], this induces additional computation and parameter setting, and a method producing good results directly is preferable.

In this paper, we demonstrate that state-of-the-art edge-aware filters can be achieved with standard Laplacian pyramids. We formulate our approach as the construction of the Laplacian pyramid of the filtered output. For each output pyramid coefficient, we render a filtered version of the full-resolution image, processed to have the desired properties according to the corresponding local image value at the same scale, build a new Laplacian pyramid from the filtered image, and then copy the corresponding coefficient to the output pyramid. The advantage of this approach is that while it may be nontrivial to produce an image with the desired property everywhere, it is often easier to obtain the property locally. For instance, global detail enhancement typically requires a nonlinear image decomposition, e.g., [Fattal et al. 2007; Farbman et al. 2008; Subr et al. 2009], but enhancing details in the vicinity of a pixel can be done with a simple S-shaped contrast curve centered on the pixel intensity. This local transformation only achieves the desired effect in the neighborhood of a pixel, but is sufficient to estimate the fine-scale Laplacian coefficient of the output. We repeat this process for each coefficient independently and collapse the pyramid to produce the final output.

We motivate this approach by analyzing its effect on step edges and show that edges can be differentiated from small-scale details with a simple threshold on color differences. We propose an algorithm that has a $\mathcal{O}(N \log N)$ complexity for an image with N pixels. While our algorithm is not as fast as other techniques, it can achieve visually compelling results hard to obtain with previous work. We demonstrate our approach by implementing a series of edge-aware filters such as edge-preserving smoothing, detail enhancement, tone mapping, and inverse tone mapping. We provide numerous results, including large-amplitude image transformations. None of them exhibit halos, thereby showing that high-quality halo-free results can be indeed obtained using only the Laplacian pyramid, which was previously thought impossible.

Contributions The main contribution of this work is a flexible approach to achieve edge-aware image processing through simple point-wise manipulation of Laplacian pyramids. Our approach builds upon a new understanding of how image edges are represented in Laplacian pyramids and how to manipulate them in a local fashion. Based on this, we design a set of edge-aware filters that produce high-quality halo-free results.

2 Related Work

Edge-aware Image Processing Edge-aware image manipulation has already received a great deal of attention and we refer to books and surveys for an in-depth presentation [Aubert and Kornprobst 2002; Kimmel 2003; Paris et al. 2009]. Recently, several methods have demonstrated satisfying results with good performance, e.g., [Chen et al. 2007; Farbman et al. 2008; Fattal 2009; Subr et al. 2009; Criminisi et al. 2010; He et al. 2010; Kass and Solomon 2010]. Our practical contribution is to provide filters that consistently achieve results at least as good, have easy-to-set parameters, can be implemented with only basic image resizing routines, are non-iterative, and do not rely on optimization or post-processing. In particular, unlike gradient-domain methods, e.g., [Fattal et al. 2002], we do not need to solve the Poisson equation which may introduce artifacts with non-integrable gradient fields. From a conceptual standpoint, our approach is based on image pyramids and is inherently multiscale, which differentiates it from methods that are expressed as a two-scale decomposition, e.g., [Chen et al. 2007; Subr et al. 2009; He et al. 2010].

Pyramid-based Edge-aware Filtering As described earlier, pyramids are not the typical representation of choice for filtering an image in an edge-preserving way, and only a few techniques along these lines have been proposed. A first approach is to directly rescale the coefficients of a Laplacian pyramid, however, this typically produces halos [Li et al. 2005]. While halos may be tolerable in the context of medical imaging, e.g., [Vuylsteke and Schoeters 1994; Dippel et al. 2002], they are unacceptable in photography.

Fattal et al. [2002] avoid halos by using a Gaussian pyramid to compute scaling factors applied to the image gradients. They reconstruct the final image by solving the Poisson equation. In comparison, our approach directly manipulates the Laplacian pyramid of the image and does not require global optimization. Fattal et al. [2007] use a multi-scale image decomposition to combine several images for detail enhancement. Their decomposition is based on repeated applications of the bilateral filter. Their approach is akin to building a Laplacian pyramid but without decimating the levels and with a spatially varying kernel instead of a Gaussian kernel. However, their study is significantly different from ours because it focuses on multi-image combination and speed. In a similar spirit, Farbman et al. [2008] compute a multi-scale edge-preserving decomposition with a least-squares scheme instead of bilateral filtering. This work also differs from ours since its main concern is the definition and application of a new optimization-based filter. In the context of tone mapping, Mantiuk et al. [2006] model human perception with a Gaussian pyramid. The final image is the result of an optimization process, which departs from our goal of working only with pyramids.

Fattal [2009] describes wavelet bases that are specific to each image. He takes edges explicitly into account to define the basis functions, thereby reducing the correlation between pyramid levels. From a conceptual point of view, our work and Fattal's are complementary. Whereas he designed pyramids in which edges do not generate correlated coefficients, we seek to better understand this correlation to preserve it during filtering.

Li et al. [2005] demonstrate a tone-mapping operator based on a generic set of spatially-invariant wavelets, countering the popular belief that such wavelets are not appropriate for edge-aware processing. Their method relies on a corrective scheme to preserve the spatial and intra-scale correlation between coefficients, and they also advocate computing each level of the pyramid at full resolution to prevent aliasing. However, when applied to Laplacian pyramids, strong corrections are required to avoid halos, which prevents a large increase of the local contrast. In comparison, in this work, we show that Laplacian pyramids can produce a wide range of edge-aware effects, including extreme detail amplification, without introducing halos.

Gaussian pyramids are closely related to the concept of Gaussian scale-space defined by filtering an image with a series of Gaussian kernels of increasing size. While these approaches are also concerned with the correlation between scales created by edges, they are used mostly for purposes of analysis, e.g., [Witkin 1983; Witkin et al. 1987].

Background on Gaussian and Laplacian Pyramids Our approach is based on standard image pyramids, whose construction we summarize briefly. For more detail, see [Burt and Adelson 1983]. Given an image I, its *Gaussian pyramid* is a set of images $\{G_\ell\}$ called *levels*, representing progressively lower resolution versions of the image, in which high-frequency details progressively disappear. In the Gaussian pyramid, the bottom-most level is the original image, $G_0 = I$, and $G_{\ell+1} = \text{downsample}(G_\ell)$ is a low-pass version of G_ℓ with half the width and height. The filtering and decimation process is iterated n times, typically until the level G_n has only a few pixels. The *Laplacian pyramid* is a closely related construct, whose levels $\{L_\ell\}$ represent details at different spatial scales, decomposing the image into roughly separate fre-

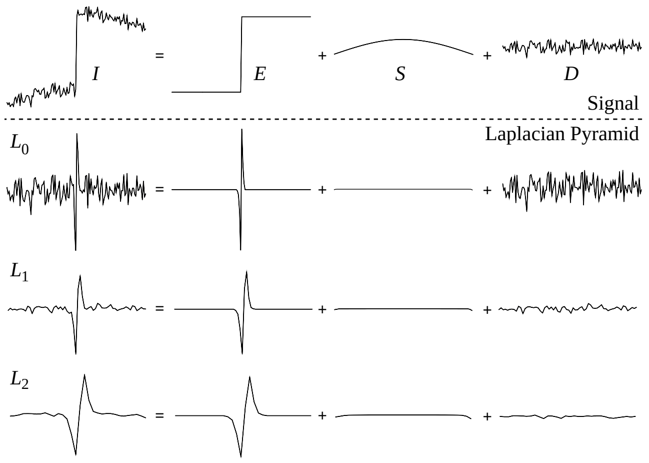


Figure 2: *The 1D signal I is decomposed into three main components: a step edge E, a slowly varying signal S, and a high-frequency detail signal D. The Laplacian pyramid representation (three levels) is shown for the signal and its individual components.*

quency bands. Levels of the Laplacian pyramid are defined by the details that distinguish successive levels of the Gaussian pyramid, $L_\ell = G_\ell - \text{upsample}(G_{\ell+1})$, where upsample$(\cdot)$ is an operator that doubles the image size in each dimension using a smooth kernel. The top-most level of the Laplacian pyramid, also called the *residual*, is defined as $L_n = G_n$ and corresponds to a tiny version of the image. A Laplacian pyramid can be *collapsed* to reconstruct the original image by recursively applying $G_\ell = L_\ell + \text{upsample}(G_{\ell+1})$ until $G_0 = I$ is recovered.

3 Dealing with Edges in Laplacian Pyramids

The goal of edge-aware processing is to modify an input signal I to create an output I', such that the large discontinuities of I, i.e., its edges, remain in place, and such that their profiles retain the same overall shape. That is, the amplitude of significant edges may be increased or reduced, but the edge transitions should not become smoother or sharper. The ability to process images in this edge-aware fashion is particularly important for techniques that manipulate the image in a spatially-varying way, such as image enhancement or tone mapping. Failure to account for edges in these applications leads to distracting visual artifacts such as haloing, shifted edges, or reversals of gradients.

To gain intuition, we first analyze 1D signals and edges using Laplacian pyramids, based on a decomposition into step edge, smoothly-varying, and detail components. As we show, each Laplacian coefficient can be attributed to either the edge or the detail component, opening up the ability to manipulate the image in an edge-aware way with simple local processing. We then consider the specific application of range compression and use our analysis to develop an edge-aware compression method based on direct point-wise processing of the original signal. Subsequently, we formalize this approach, generalize it to other edge-aware effects, and extend it to handle 2D images with gray-scale or color pixels.

3.1 Components of One-dimensional Signals

In this section, we show that Laplacian coefficients can be mostly attributed to the details or to the edges. This decomposition will be the starting point of the construction of edge-aware filters in Section 3.2. We represent a 1D signal I as a Laplacian pyramid $\{L_\ell\}$, depicted on the left side of Figure 2, and begin our analysis by looking at how Laplacian coefficients describe this signal. We start with the first level L_0, encoding the highest-frequency variations of I at the scale of 1 or 2 pixels. As can be seen in Figure 2, L_0 captures high-frequency details away from large discontinuities, and in their immediate vicinity, fine-scale detail is dominated by the discontinuity's influence. Each of the L_0 coefficients is concerned only by what happens in a small window that spans only a few surrounding pixels. For now, we assume that there is only a single discontinuity within this window.

For our analysis, we decompose the local signal I as the sum of three intuitive components: a step edge E, a detail layer D, and a slowly varying signal S (Fig. 2 top). E represents the discontinuity and is defined as $E(x) = 0$ if $x < 0$ and $E(x) = A$ otherwise. D represents the high-frequency texture, which has zero mean. Since we assume a single discontinuity in the neighborhood, its amplitude is significantly smaller than E, i.e., $\max|D| \ll A$. S represents the lower-frequency component of the signal such as a constant slope or large-scale smooth oscillations.

Local Edge-Detail Separation First, we show that each of the Laplacian coefficients in $L_0[I]$ can be attributed almost fully to a single component, either the step edge or the detail layer. Away from the edge, E is flat and $L_0[E] = 0$. At the edge, $G_0[E] = A/2$ and $|L_0[E]| = A/2$ because of the symmetry around $x = 0$. Because D is high-frequency, $G_0[D] \approx 0$ and $L_0[D] \approx D$. Finally, $L_0[S] \approx 0$ because it is low-frequency. Since building a Laplacian pyramid is a linear operation, we have $L_0[I] \approx L_0[E]$ near the edge and $L_0[I] \approx L_0[D]$ away from it (Fig. 2).

Edge-Detail Separation for the Full Pyramid So far, our analysis has focused on the first pyramid level L_0. Going to coarser-scale higher levels has several effects, but these do not affect our key result that every Laplacian coefficient can be attributed to either a step edge or the detail layer. First, at higher levels of the pyramid, high-frequency details progressively disappear, since they have already been captured by the lower levels (Fig. 2). Secondly, because of decimation, smooth variations become higher frequency. As a result, the highest-frequency part of S may generate new details or new edges that are treated consistently with our above analysis, meaning that the effects of low-frequency edges can ultimately be attributed to either a step edge or detail layer at some scale. Finally, going up in the pyramid also affects step discontinuities. The amplitude and profile of E are not affected because the smoothing effect is canceled by decimation. But other edges can appear, either because of the low-frequency component or because the distance between adjacent edges is reduced by decimation. In each of these cases, discontinuities still dominate the Laplacian coefficients and we can still identify them correctly as edges.

Discussion If there are two discontinuities, as in the case of a double step edge due to a line profile, similar arguments let us reach the equivalent conclusion that the double edge dominates in its vicinity and that detail coefficients take over away from it. If there are more discontinuities, we are actually observing a high-amplitude texture that dominates the signal everywhere. Large variations like these are treated as edges in most applications, including ours, so processing them will increase or decrease their amplitude in the same way that isolated edges are handled.

3.2 Application to Range Compression

Our analysis has shown that each Laplacian coefficient can be either attributed to the details or to an edge. We first exploit this result for edge-aware range compression on toy 1D examples in Figures 3 and 4, processing the signal so that the amplitude of its edges is reduced while its details are preserved. Then, we formally justify

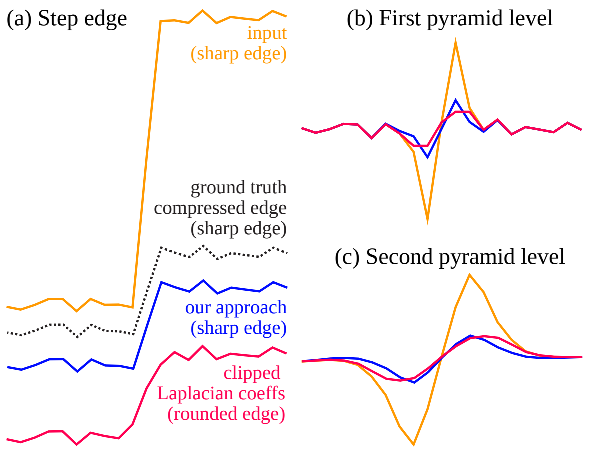


Figure 3: *Range compression applied to a step edge with fine details (a). The different versions of the edge are offset vertically so that their profiles are clearly visible. Truncating the Laplacian coefficients smooths the edge (red), an issue which Li et al. [2005] have identified as a source of artifacts in tone mapping. In comparison, our approach (blue) preserves the edge sharpness and very closely reproduces the desired result (black). Observing the shape of the first two levels (b,c) shows that clipping the coefficients significantly alters the shape of the signal (red vs. orange). The truncated coefficients form wider lobes whereas our approach produces profiles nearly identical to the input (blue vs. orange).*

this approach with the principle of edge-detail separation that we exposed in the previous section. This example also serves as an introduction to our more general approach for edge-aware image processing (Sec. 4).

Pyramid-based Range Compression Before describing our range compression method, we discuss a few alternatives that are also based on pyramid manipulations but do not lead to satisfying results. A naïve approach to range compression is to scale down the signal uniformly, which is equivalent to scaling all the pyramid coefficients by the same factor. However, this attenuates the details and produces dull results when applied to images. A more sophisticated approach is to remark that edges correspond to large pyramid coefficients and selectively reduce these large values. This is the gist of the approach by Li et al. [2005]. For the sake of illustration, we simply threshold the large-magnitude coefficients; while this mostly achieves the desired result, the output signal (Fig. 3, red curve) is somewhat deformed near edges. Such deformations typically affect only a small band of pixels, manifested in images as thin but unsightly rim halos. Li et al. address this issue with smooth activity maps as a way to alter the coefficients while preserving their profile, and show that wavelet filters are more suitable than Laplacian pyramids for implementing this correction.

Incorporating Local Processing Motivated by our previous analysis, our approach to range compression also takes a threshold-based view of edges. However, unlike Li et al. [2005], we demonstrate how to exploit a simple edge-detail decomposition directly, without further corrections and without using a more complex wavelet-based decomposition. The key to our approach is *local processing*, which better lets us assign responsibility for each Laplacian pyramid coefficient to the edge and detail components of the signal. We formulate our approach as the construction of the pyramid $\{L[I']\}$ of the output signal I', coefficient by coefficient, reminiscent of backwards mapping for image warping.

For each pyramid coefficient, described by location and pyramid level (x_0, ℓ_0), we render an intermediate signal $\tilde{I}$, that is generated by point-wise manipulation of the full-resolution input I, based on the corresponding local signal value g_0. We take g_0 to be the coefficient of the Gaussian pyramid at the same location and level, $g_0 = G_{\ell_0}(x_0)$. For range compression, our point-wise manipulation consists of clipping high and low values of the original signal that are more than σ_r away from the local value, $\tilde{I} = \min(\max(I, g_0 - \sigma_r), g_0 + \sigma_r)$. Then, the Laplacian pyramid of this intermediate signal is computed and the coefficient $L_{\ell_0}[\tilde{I}](x_0)$ is copied to the output pyramid. Figure 3 shows the result of applying this approach on a step-like signal I of amplitude A with parameter value $\sigma_r < A$. Each destination pyramid coefficient is generated with point-wise processing corresponding to the local value, however, because this processing is applied to the whole original signal, our method effectively takes into account *global* correlations between coefficients of the Laplacian pyramid as well.

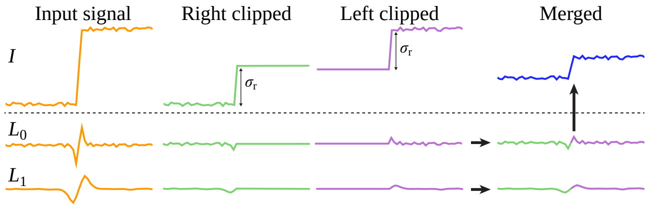


Figure 4: *Simple view of our range compression approach, which is based on thresholding and local processing. For a step-like signal similar to the one in Figure 3, our method effectively builds two Laplacian pyramids, corresponding to clipping the input based on the signal value to the left and right of the step edge, then merging their coefficients as indicated by the color coding.*

For more intuition behind this local processing approach, we illustrate the steps of the computation in Figure 4. For coefficients on the left side of the step edge, corresponding to a roughly constant local signal g_{left}, the right side of the signal will effectively be clipped at $g_{\text{left}} + \sigma_r$. Similarly, coefficients on the right side of the step edge, with a local signal g_{right}, will cause the left side of the signal to clip at $g_{\text{right}} - \sigma_r$. For this simple example, our method reduces to clipping the input signal at two different thresholds, generating Laplacian pyramids for each, then merging the pyramids based on which side of the step edge each destination coefficient corresponds to. More generally, our method entails "rendering" Laplacian pyramids in this way for every coefficient in the destination pyramid.

Analysis To understand how clipping achieves range compression, we analyze the signals involved component by component. In the final result I', clipping produces a smaller step edge with amplitude σ_r, but preserves the details, whose coefficients represent local information. Formally, the input signal I is equal to $E + D$. On the left side, the clipped signal $\tilde{I}$ is equal to $\tilde{E}_- + \tilde{D}_-$ where $\tilde{E}_-$ is a step edge of amplitude σ_r and $\tilde{D}_-$ is equal to D for $x < 0$ and 0 otherwise. Using our analysis, for $x_0 < 0$, we have $L_{\ell_0}[I'](x_0) = L_{\ell_0}[D](x_0)$ far away from the edge and $L_{\ell_0}[I'](x_0) \approx L_{\ell_0}[\tilde{E}_-](x_0)$. The situation on the right side is symmetric except that step edge $\tilde{E}_+$ differs from $\tilde{E}_-$ by a constant and in sign, which has no influence on the Laplacian coefficients. We introduce E' a step edge of amplitude σ_r and remark that $L_{\ell_0}[\tilde{E}_\pm] = L_{\ell_0}[E']$. Put together, we have $L_{\ell_0}[I'] \approx L_{\ell_0}[E' + D]$, which is the result that we seek: we have built a signal I' with the same details as I but with a smaller discontinuity.

For the application of range compression, our method uses clipping to generate the intermediate image $\tilde{I}$. However, exactly the same approach can be used with other point-wise remapping functions. All of the previous analysis remains valid provided the edges have a larger amplitude than the details. To satisfy this basic requirement, it is sufficient for the remapping function to be monotonically increasing. This is compatible with common image editing appli-

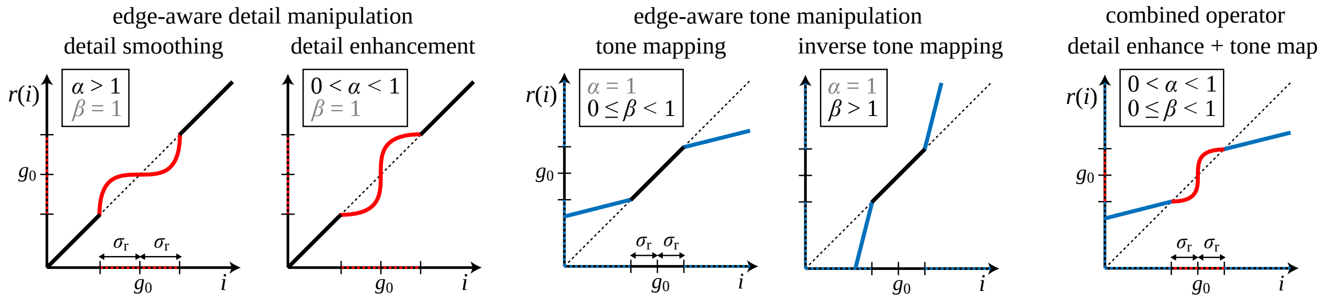


Figure 6: *Family of point-wise functions for edge-aware manipulation described in Secs. 5.2 and 5.3. The parameters α and β let us control how detail and tone are processed respectively. To compute a given Laplacian coefficient in the output, we filter the original image point-wise using a nonlinear function $r(i)$ of the form shown. This remapping function is parametrized by the Gaussian pyramid coefficient g_0, describing the local image content, and a threshold σ_r used to distinguish fine details (red) from larger edges (blue).*

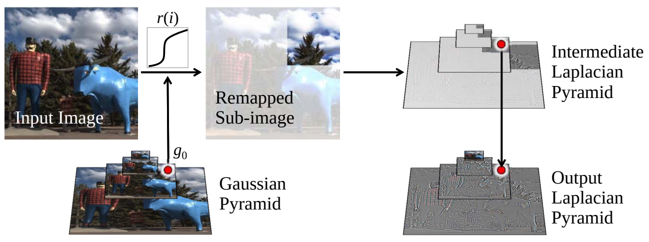


Figure 5: *Overview of the basic idea of our approach. For each pixel in the Gaussian pyramid of the input (red dot), we look up its value g_0. Based on g_0, we remap the input image using a point-wise function, build a Laplacian pyramid from this intermediate result, then copy the appropriate pixel into the output Laplacian pyramid. This process is repeated for each pixel over all scales until the output pyramid is filled, which is then collapsed to give the final result. For more efficient computation, only parts of the intermediate pyramid need to be generated.*

cations, such as contrast enhancement and reduction, which rely on the local application of such monotonic remapping functions.

4 Local Laplacian Filtering

Given this background, we are ready to introduce *Local Laplacian Filtering*, our new method for edge-aware image processing based on the Laplacian pyramid. This method draws on our previous analysis of 1D signals (Sec. 3.1) and directly generalizes the local point-wise method that we developed for 1D range compression (Sec. 3.2). A visual overview is given in Figure 5 and the pseudo-code is provided in Algorithm 1.

In Local Laplacian Filtering an input image is processed by constructing the Laplacian pyramid $\{L[I']\}$ of the output, one coefficient at a time. For each coefficient (x_0, y_0, ℓ_0), we generate an intermediate image $\tilde{I}$ by applying a point-wise monotonic remapping function $r_{g_0,\sigma_r}(\cdot)$ to the original full-resolution image. This remapping function, whose design we discuss later, depends on the local image value from the Gaussian pyramid $g_0 = G_{\ell_0}(x_0, y_0)$ and the user parameter σ_r which is used to distinguish edges from details. We compute the pyramid for the intermediate image $\{L[\tilde{I}]\}$ and copy the corresponding coefficient to the output $\{L[I']\}$. After all coefficients of the output pyramid have been computed, we collapse the output pyramid to get the final result.

Algorithm 1 $\mathcal{O}(N \log N)$ version of Local Laplacian Filtering

```
input: image $I$, parameter $\sigma_r$, remapping function $r$
output: image $I'$
  compute input Gaussian pyramid $\{G[I]\}$
  for all $(x_0, y_0, \ell_0)$ do
    $g_0 \leftarrow G_{\ell_0}(x_0, y_0)$
    determine sub-region $R_0$ needed to evaluate $L_{\ell_0}(x_0, y_0)$
    apply remapping function: $\tilde{R}_0 \leftarrow r_{g_0,\sigma_r}(R_0)$
    compute sub-pyramid $\{L_{\ell_0}[\tilde{R}_0]\}$
    update output pyramid: $L_{\ell_0}[I'](x_0, y_0) \leftarrow L_{\ell_0}[\tilde{R}_0](x_0, y_0)$
  end for
  collapse output pyramid: $I' \leftarrow \text{collapse}(\{L_\ell[I']\})$
```

A direct implementation of this algorithm yields a complexity in $\mathcal{O}(N^2)$ with N being the number of pixels in the image, since each coefficient entails the construction of another pyramid with $\mathcal{O}(N)$ pixels. However, this cost can be reduced in a straightforward way by processing only the sub-pyramid needed to evaluate $L_{\ell_0}[\tilde{I}](x_0, y_0)$, illustrated in Figure 5. The base of this sub-pyramid lies within a $K \times K$ sub-region R_0 of the input image I, where $K = \mathcal{O}(2^{\ell_0})$; for Laplacian pyramids built using a standard 5-tap interpolation filter, it can be shown that $K = 3(2^{\ell_0+2} - 1)$. Put together with the fact that level ℓ_0 contains $\mathcal{O}(N/2^{\ell_0})$ coefficients, each level requires the manipulation of $\mathcal{O}(N)$ coefficients in total. Since there are $\mathcal{O}(\log N)$ levels in the pyramid, the overall complexity of our algorithm is $\mathcal{O}(N \log N)$. Later we will see that some applications only require a fixed number of levels to be processed or limit the depth of the sub-pyramids to a fixed value, reducing the complexity of our algorithm further.

Remapping Function for Gray-scale Images We assume the user has provided a parameter σ_r such that intensity variations smaller than σ_r should be considered fine-scale details and larger variations are edges. As a center point for this function we use $g_0 = G_{\ell_0}(x_0, y_0)$, which represents the image intensity at the location and scale where we compute the output pyramid coefficient. Intuitively, pixels closer than σ_r to g_0 should be processed as details and those farther than σ_r away should be processed as edges. We differentiate their treatment by defining two functions r_d and r_e, such that $r(i) = r_d(i)$ if $|i - g_0| \leq \sigma_r$ and $r(i) = r_e(i)$ otherwise. Since we require r to be monotonically increasing, r_d and r_e must have this property as well. Furthermore, to avoid the creation of spurious discontinuities, we constrain r_d and r_e to be continuous by requiring that $r_d(g_0 \pm \sigma_r) = r_e(g_0 \pm \sigma_r)$.

The function r_d modifies the fine-scale details by altering the oscillations around the value g_0. In our applications we process positive and negative details symmetrically, letting us write:

$$r_d(i) \;=\; g_0 + \text{sign}(i - g_0)\, \sigma_r\, f_d(|i - g_0|/\sigma_r) \tag{1}$$

where f_d is a smooth function mapping from $[0,1]$ to $[0,1]$ that controls how details are modified. The advantage of this formulation is that it depends only on the amplitude of the detail $|i - g_0|$ relative

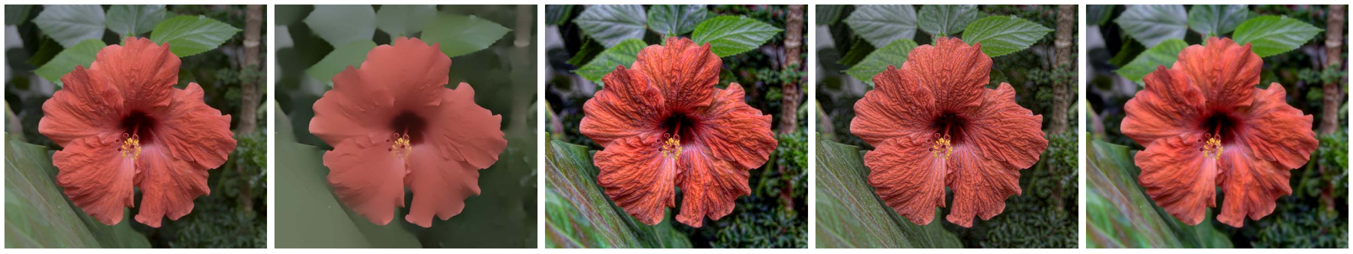

(a) input (b) reduced details ($\alpha = 4$) (c) increased details (all levels, $\alpha = 0.25$) (d) increased details (lowest 2 levels, $\alpha = 0.25$) (e) increased details (level 3 and higher, $\alpha = 0.25$)

Figure 7: *Smoothing and enhancement of detail, while preserving edges ($\sigma_r = 0.3$). Processing only a subset of the levels controls the frequency of the details that are manipulated (c,d,e). The images have been cropped to make the flower bigger and its details more visible.*

to the parameter σ_r, i.e., $|i - g_0|/\sigma_r = 1$ corresponds to a detail of maximum amplitude according to the user-defined parameter. Analogously, r_e is a function that modifies the amplitude of edges that we again formulate in a symmetric way:

$$r_e(i) \ = \ g_0 + \text{sign}(i - g_0)\big(f_e(|i - g_0| - \sigma_r) + \sigma_r\big) \qquad (2)$$

where f_e is a smooth non-negative function defined over $[0, \infty)$. In this formulation, r_e depends only on the amplitude above the user threshold σ_r, i.e., $|i - g_0| - \sigma_r$. The function f_e controls how the edge amplitude is modified since an edge of amplitude $\sigma_r + a$ becomes an edge with amplitude $\sigma_r + f_e(a)$. For our previous 1D range compression example, clipping edges corresponds to $f_e = 0$, which limits the amplitude of all edges to σ_r. Useful specific choices for r_d and r_e are described in the next section and are illustrated in Figure 6.

The advantage of the functional forms defined in Equations 1 and 2 is that they ensure that r is continuous and increasing, and the design of a specific filter boils down to defining the two point-wise functions f_d and f_e that each have clear roles: f_d controls the amplification or attenuation of details while f_e controls the amplification or attenuation of edges.

Extension to Color Images To handle color, it is possible to treat only the luminance channel and reintroduce chrominance after image processing (Sec. 5.3). However, our approach extends naturally to color images as well, letting us deal directly with 3D vectors representing, e.g., the RGB or CIE-Lab channels. Algorithm 1 still applies, and we need only to update r_d and r_e, using bold typeface to indicate vectors:

$$r_d(\mathbf{i}) \ = \ \mathbf{g}_0 + \text{unit}(\mathbf{i} - \mathbf{g}_0)\,\sigma_r\, f_d(\|\mathbf{i} - \mathbf{g}_0\|/\sigma_r) \qquad (3a)$$

$$r_e(\mathbf{i}) \ = \ \mathbf{g}_0 + \text{unit}(\mathbf{i} - \mathbf{g}_0)\big[f_e(\|\mathbf{i} - \mathbf{g}_0\| - \sigma_r) + \sigma_r\big] \qquad (3b)$$

with $\text{unit}(\mathbf{v}) = \mathbf{v}/\|\mathbf{v}\|$ if $\mathbf{v} \neq \mathbf{0}$ and $\mathbf{0}$ otherwise. These equations define details as colors within a ball of radius σ_r centered at $\mathbf{g}_0$ and edges as the colors outside it. They also do not change the roles of f_d and f_e, letting the same 1D functions that modify detail and edges in the gray-scale case be applied to generate similar effects in color. For images whose color channels are all equal, these formulas reduce to the gray-scale formulas of Eq. 1 and 2.

5 Applications and Results

We now demonstrate how to realize practical image processing applications using our approach and discuss implementation details. First we address edge-preserving smoothing and detail enhancement, followed by tone mapping and related tools. We validate our method with images used previously in the literature [Fattal et al. 2002; Durand and Dorsey 2002; Paris et al. 2009; Farbman et al. 2008; Fattal 2009] and demonstrate that our method produces artifact-free results.

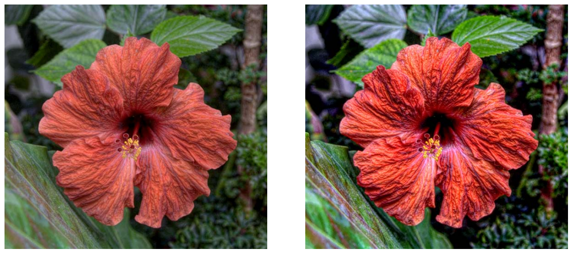

(a) $\sigma_r = 0.2$ (b) $\sigma_r = 0.5$

Figure 8: *Effect of the σ_r parameter for detail enhancement ($\alpha = 0.25$). Same input as Figure 7.*

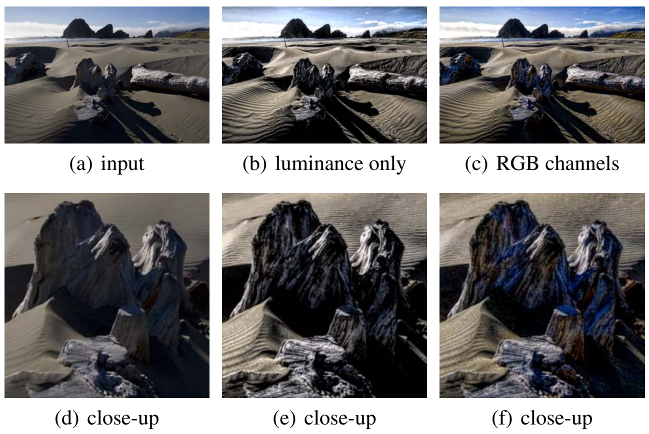
5.1 Implementation

We use the pyramids defined by Burt and Adelson [1983], based on 5×5 kernels. On a 2.26 GHz Intel Xeon CPU, we process a one-megapixel image in about a minute using a single thread. This can be halved by limiting the depth of the intermediate pyramid to at most 5 levels, by applying the remapping to level $\max(0, \ell_0 - 3)$ rather than always starting at the full-resolution image. This amounts to applying the remapping to a downsampled version of the image when processing coarse pyramid levels. The resulting images are visually indistinguishable from the full-pyramid process with a PSNR on the order of 30 to 40 dB. While this performance is slower than previous work, our algorithm is highly data parallel and can easily exploit a multi-core architecture. Using OpenMP, we obtain a $8\times$ speed-up on an 8-core machine, bringing the running time down to 4 seconds.

5.2 Detail Manipulation

To modify the details of an image we define an S-shaped point-wise function, as is classically used for the local manipulation of contrast. For this purpose, we use a power curve $f_d(\Delta) = \Delta^\alpha$, where $\alpha > 0$ is a user-defined parameter. Values larger than 1 smooth the details out, while values smaller than 1 increase their contrast (Figs. 6 and 7). To restrict our attention to the details of an image, we set the edge-modifying function to the identity $f_e(a) = a$.

In the context of detail manipulation, the parameter σ_r controls how at what magnitude signal variations should be considered edges and therefore be preserved. Large values allow the filter to alter larger portions of the signal and yield larger visual changes (Fig. 8). In its basic form detail manipulation is applied at all scales, but one can also control which scales are affected by limiting processing to a subset of the pyramid levels (Figs. 7c,d,e). While this control is discrete, the changes are gradual, and one can interpolate between the results from two subsets of levels if continuous control is desired. Our results from Figures 7 and 8 are comparable to results


(a) input (b) luminance only (c) RGB channels

(d) close-up (e) close-up (f) close-up

Figure 9: *Filtering only the luminance (b) preserves the original colors in (a), while filtering the RGB channels (c) also modifies the color contrast ($\alpha = 0.25$, $\beta = 1$, $\sigma_r = 0.4$).*

of Farbman et al. [2008], however, we do not require the complex machinery of a multi-resolution preconditioned conjugate gradient solver. Note that our particular extension to color images allows us to boost the color contrast as well (Figs. 7, 8, and 9).

Reducing Noise Amplification As in other techniques for texture enhancement, increasing the contrast of the details may make noise and artifacts from lossy image compression more visible. We mitigate this issue by limiting the smallest Δ amplified. In our implementation, when $\alpha < 1$, we compute $f_d(\Delta) = \tau\Delta^\alpha + (1-\tau)\Delta$ where τ is a smooth step function equal to 0 if Δ is less than 1% of the maximum intensity, 1 if it is more than 2%, with a smooth transition in between. All the results in this paper and supplemental material are computed with this function.

5.3 Tone Manipulation

Our approach can also be used for reducing the intensity range of a HDR image, according to the standard tone mapping strategy of compressing the large-scale variations while preserving (or enhancing) the details [Tumblin and Turk 1999]. In our framework, we manipulate large-scale variations by defining a point-wise function modifying the edge amplitude, $f_e(a) = \beta a$, where $\beta \geq 0$ is a user-defined parameter (Fig. 6).

In our implementation of tone manipulation we process the image intensity channel only and keep the color unchanged [Durand and Dorsey 2002]. We compute an intensity image $I_i = \frac{1}{61}(20I_r + 40I_g + I_b)$ and color ratios $(\rho_r, \rho_g, \rho_b) = \frac{1}{I_i}(I_r, I_g, I_b)$ where I_r, I_g, and I_b are the RGB channels. We apply our filter on the log intensities $\log(I_i)$ [Tumblin and Turk 1999], using the natural logarithm. For tone mapping, we set our filter with $\alpha \leq 1$ so that details are preserved or enhanced, and $\beta < 1$ so that edges are compressed. This produces new values $\log(I'_i)$, which we must then map to the displayable range of $[0, 1]$. We remap the result $\log(I'_i)$ by first offsetting its values to make its maximum 0, then scaling them so that they cover a user-defined range [Durand and Dorsey 2002; Li et al. 2005]. In our implementation, we estimate a robust maximum and minimum with the 99.5th and 0.5th percentiles, and we set the scale factor so that the output dynamic range is $100:1$ for the linear intensities. Finally, we multiply the intensity by the color ratios (ρ_r, ρ_g, ρ_b) to obtain the output RGB channels, then gamma correct with an exponent of $1/2.2$ for display. We found that fixing the output dynamic range makes it easy to achieve a consistent look but also constrains the system. As a result, the σ_r and β parameters have similar effects, both controlling the balance between local and global contrast in the rendered image (Fig. 10). From a practical standpoint, we advise keeping σ_r fixed and varying the slope β between 0, where the local contrast is responsible for most of the dynamic range, and 1, where the global contrast dominates. Unless otherwise specified, we use $\sigma_r = \log(2.5)$, which gave consistently good results in our experiments. Since we work in the log domain, this value corresponds to a ratio between pixel intensities. It does not depend on the dynamic range of the scene, and assumes only that the input HDR image measures radiance up to scale.

Our tone mapping operator builds upon standard elements from previous work that could be substituted for others. For instance, one could instead use a sigmoid to remap the intensities to the display

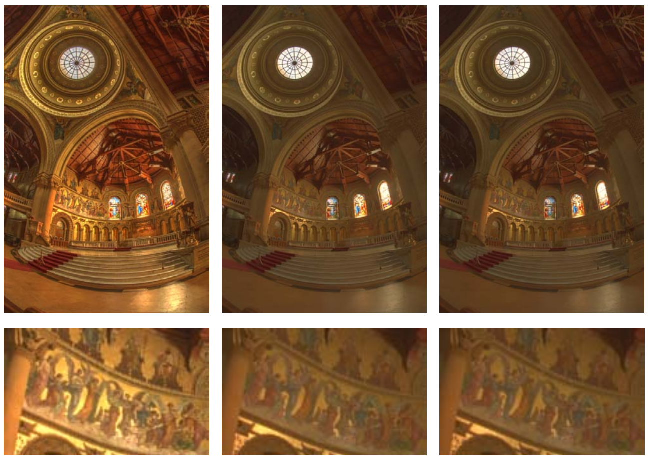
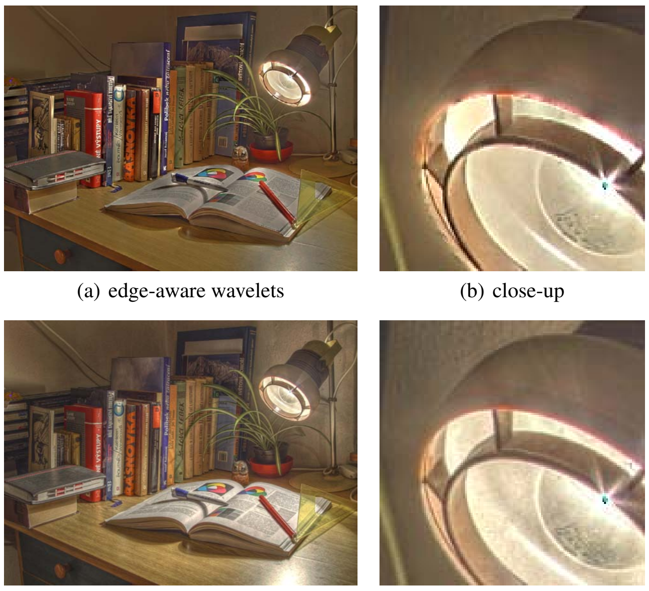
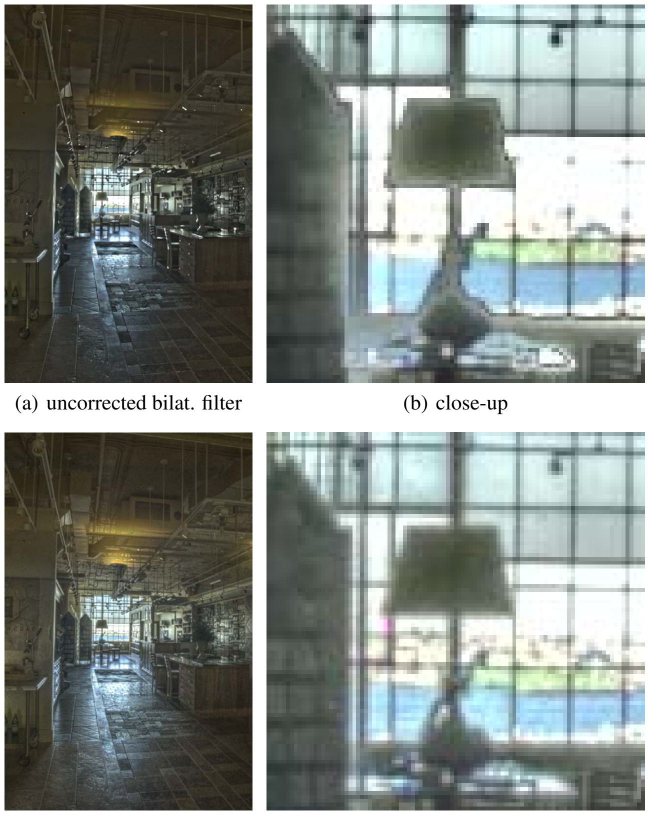

(a) $\beta = 0$ $\sigma_r = \log(2.5)$ (b) $\beta = 0$ $\sigma_r = \log(30)$ (c) $\beta = 0.75$ $\sigma_r = \log(2.5)$

Figure 10: *β and σ_r have similar effects on tone mapping results, they control the balance between global and local contrast. α is set to 1 in all three images.*


(a) edge-aware wavelets (b) close-up

(c) our result (d) close-up

Figure 11: *The extreme contrast near the light bulb is particularly challenging. Images (a) and (b) are reproduced from [Fattal 2009]. The edge-aware wavelets suffer from aliasing and generate an irregular edge (b). In comparison, our approach (d) produces a clean edge. We set our method to approximately achieve the same level of details ($\sigma_r = \log(3.5)$, $\alpha = 0.5$, $\beta = 0$).*


(a) uncorrected bilat. filter (b) close-up

(c) our result (d) close-up

Figure 12: *The bilateral filter sometimes over-sharpens edges, which can leads to artifacts (b). We used code provided by [Paris and Durand] and multiplied the detail layer by 2.5 to generate these results. Although such artifacts can be fixed in post-processing, this introduces more complexity to the system and requires new parameters. Our approach produces clean edges directly (d). We set our method to achieve approximately the same visual result ($\sigma_r = \log(2.5)$, $\alpha = 0.5$, $\beta = 0$).*

range [Reinhard et al. 2002] or use a different color management method, e.g., [Mantiuk et al. 2009]. Also, we did not apply any additional "beautifying curve" or increased saturation as is commonly done in photo editing software. Our approach produces a clean output image that can be post-processed in this way if desired.

Range compression is a good test case to demonstrate the abilities of our pyramid-based filters because of the large modification involved. For high compression, even subtle inaccuracies can become visible, especially at high-contrast edges. In our experiments, we did not observe aliasing or over-sharpening artifacts even on cases where other methods suffer from them (Figs. 11 and 12). We also stress-tested our operator by producing results with a low global contrast ($\beta = 0$) and high local details ($\alpha = 0.25$). In general, the results produced by our method did not exhibit any particular problems (Fig. 17). We compare exaggerated renditions of our method with Farbman et al. [2008] and Li et al. [2005]. Our method produces consistent results without halos, whereas the other methods either create halos or fail to exaggerate detail (Fig. 15).

One typical difficulty we encountered is that sometimes the sky interacts with other elements to form high-frequency textures that undesirably get amplified by our detail-enhancing filter (Figs. 9b and 13). Such "misinterpretation" is common to all low-level filters without semantic understanding of the scene, and typically requires user feedback to correct [Lischinski et al. 2006].

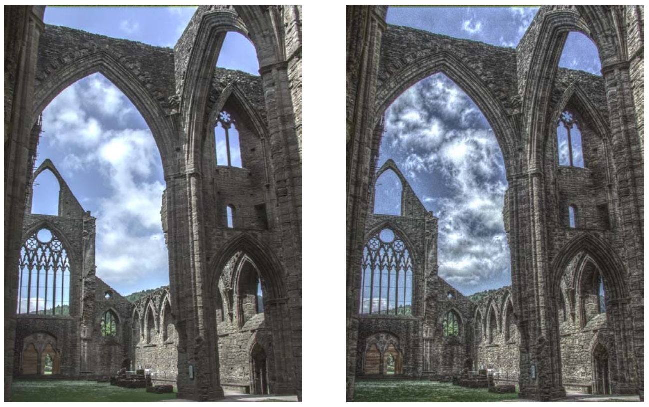
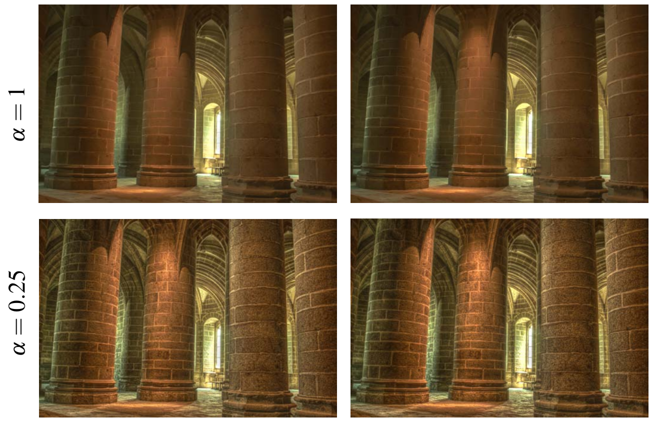

(a) no detail increase ($\alpha = 1$) (b) detail increased ($\alpha = 0.25$)

Figure 13: *Our approach is purely signal-based and its ignorance of scene semantics can lead to artifacts. For a large increase in local contrast (b), at a level similar to Figure 17, the sky gets locally darker behind clouds, because it forms a blue-white texture amplified by our filter. Our result for this example is good elsewhere, and this issue does not appear with a more classical rendition (a).*

We also experimented with inverse tone mapping, using slope values β larger than 1 to increase the dynamic range of a normal image. Since we operate on log intensities, roughly speaking the linear dynamic range gets exponentiated by β. Applying our tone-mapping operator on these range-expanded results gives images close to the originals, typically with a PSNR between 25 and 30 dB for $\beta = 2.5$. This shows that our inverse tone mapping preserves the image content well. While a full-scale study on an HDR monitor is beyond the scope of our paper, we believe that our simple approach can complement other relevant techniques, e.g., Masia et al. [2009]. Sample HDR results are provided in supplemental material.

5.4 Discussion

While our method can fail in the presence of excessive noise or when extreme parameter settings are used (e.g., the *lenna* picture in supplemental material has a high level of noise), we found that our filters are very robust and behave well over a broad range of settings. Figure 16 shows a variety of parameters values applied to the same image and the results are consistently satisfying, high-quality, and halo-free; many more such examples are provided in



(a) full resolution (1840×1224) (b) preview (500×333)

Figure 14: *Our approach generates faithful previews when applied to a low-resolution version of an image ($\beta = 0$, $\sigma_r = \log(2.5)$).*

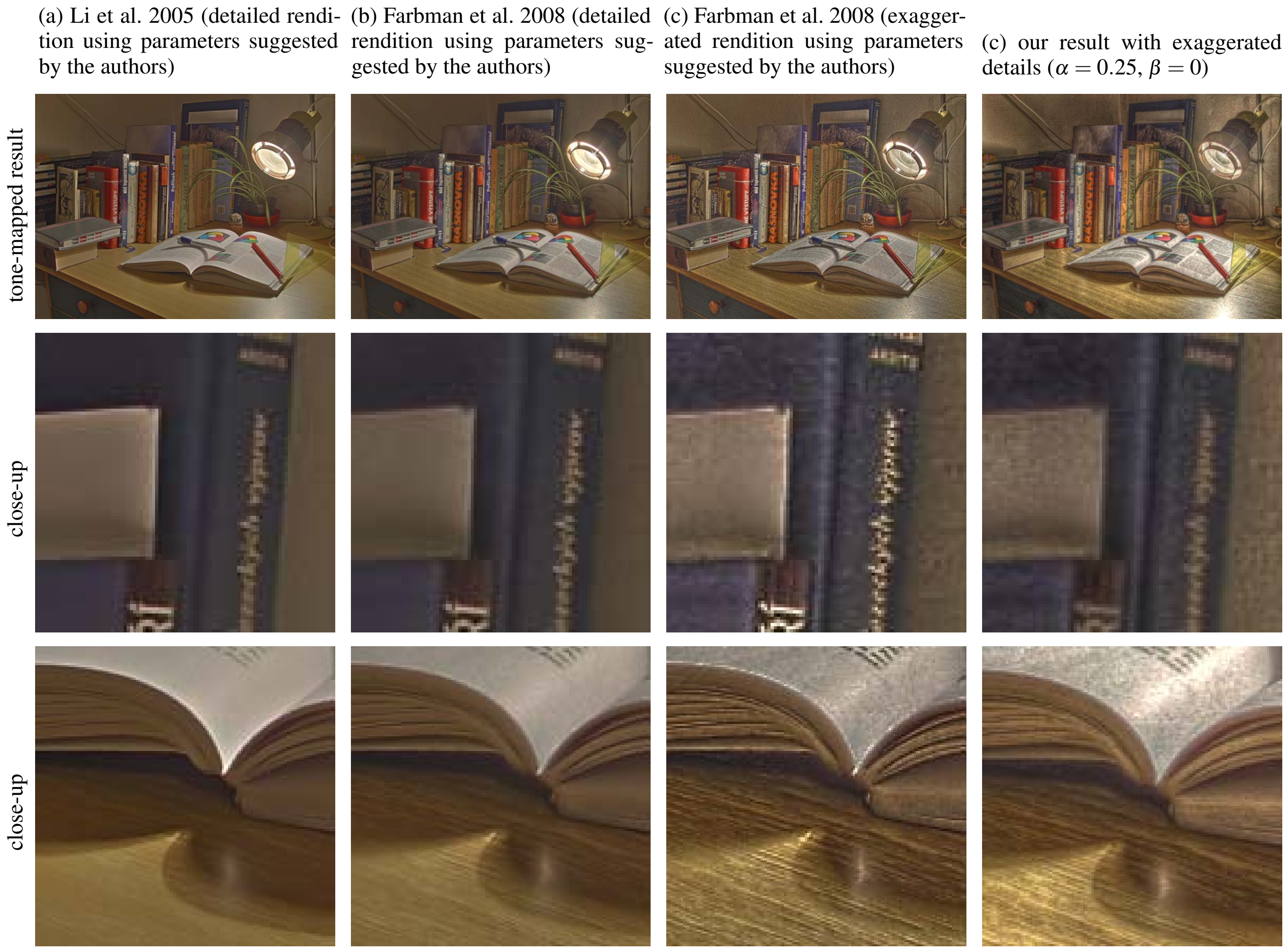

(a) Li et al. 2005 (detailed rendition using parameters suggested by the authors)
(b) Farbman et al. 2008 (detailed rendition using parameters suggested by the authors)
(c) Farbman et al. 2008 (exaggerated rendition using parameters suggested by the authors)
(c) our result with exaggerated details ($\alpha = 0.25$, $\beta = 0$)
tone-mapped result
close-up
close-up

Figure 15: *We compare exaggerated, tone-mapped renditions of an HDR image. The wavelet-based method by Li et al. [2005] is best suited for neutral renditions and generates halos when one increases the level of detail (a). The multi-scale method by Farbman et al. [2008] performs better and produces satisfying results for intermediate levels of detail (b), but halos and edge artifacts sometimes appear for a larger increase, as in this image for instance; see the edge of the white square on the blue book cover and the edge of the open book (c). In comparison, our approach achieves highly detailed renditions without artifacts (d). These results as well as many others may be better seen in the supplemental material.*

supplemental material. While the goal of edge-aware processing can be ill-defined, the results that we obtain shows that our approach allows us to realize many edge-aware effects with intuitive parameters and a simple implementation. The current shortcoming of our approach is its running time. We can mitigate this issue thanks to the multi-scale nature of our algorithm, allowing us to generate quick previews that are faithful to the full resolution results (Fig. 14). Furthermore, the algorithm is highly parallelizable and should lend itself to a fast GPU implementation. Beyond these practical aspects, our main contribution is a better characterization of the multi-scale properties of images. Many problems related to photo editing are grounded in these properties of images and we believe that a better understanding can have benefits beyond the applications demonstrated in this paper.

6 Conclusion

We have presented a new technique for edge-aware image processing based solely on the Laplacian pyramid. It is conceptually simple, allows for a wide range of edge-aware filters, and consistently produces artifact-free images. We demonstrate high-quality results over a large variety of images and parameter settings, confirming the method's robustness. Our results open new perspectives on multi-scale image analysis and editing since Laplacian pyramids were previously considered as ill-suited for manipulating edges. Given the wide use of pyramids and the need for edge-aware processing, we believe our new insights can have a broad impact in the domain of image editing and its related applications.

Acknowledgments We would like to thank Ted Adelson, Bill Freeman and Frédo Durand for inspiring discussions and encouragement, Alan Erickson for the *Orion* image, and the anonymous reviewers for their constructive comments. This work was supported in part by an NSERC Postdoctoral Fellowship, the Quanta T-Party, NGA NEGI-1582-04-0004, MURI Grant N00014-06-1-0734, and gifts from Microsoft, Google and Adobe. We thank Farbman et al. and Li et al. for their help with comparisons.

References

AUBERT, G., AND KORNPROBST, P. 2002. *Mathematical prob-*

lems in image processing: Partial Differential Equations and the Calculus of Variations, vol. 147 of *Applied Mathematical Sciences*. Springer.

BAE, S., PARIS, S., AND DURAND, F. 2006. Two-scale tone management for photographic look. *ACM Transactions on Graphics (Proc. SIGGRAPH) 25*, 3, 637–645.

BHAT, P., ZITNICK, C. L., COHEN, M., AND CURLESS, B. 2010. Gradientshop: A gradient-domain optimization framework for image and video filtering. *ACM Transactions on Graphics 29*, 2.

BUADES, A., COLL, B., AND MOREL, J.-M. 2006. The staircasing effect in neighborhood filters and its solution. *IEEE Transactions on Image Processing 15*, 6, 1499–1505.

BURT, P. J., AND ADELSON, E. H. 1983. The Laplacian pyramid as a compact image code. *IEEE Transactions on Communication 31*, 4, 532–540.

CHEN, J., PARIS, S., AND DURAND, F. 2007. Real-time edge-aware image processing with the bilateral grid. *ACM Transactions on Graphics (Proc. SIGGRAPH) 26*, 3.

CRIMINISI, A., SHARP, T., ROTHER, C., AND PEREZ, P. 2010. Geodesic image and video editing. *ACM Transactions on Graphics 29*, 5.

DIPPEL, S., STAHL, M., WIEMKER, R., AND BLAFFERT, T. 2002. Multiscale contrast enhancement for radiographies: Laplacian pyramid versus fast wavelet transform. *IEEE Transactions on Medical Imaging 21*, 4.

DURAND, F., AND DORSEY, J. 2002. Fast bilateral filtering for the display of high-dynamic-range images. *ACM Transactions on Graphics (Proc. SIGGRAPH) 21*, 3.

FARBMAN, Z., FATTAL, R., LISCHINSKI, D., AND SZELISKI, R. 2008. Edge-preserving decompositions for multi-scale tone and detail manipulation. *ACM Transactions on Graphics (Proc. SIGGRAPH) 27*, 3.

FATTAL, R., LISCHINSKI, D., AND WERMAN, M. 2002. Gradient domain high dynamic range compression. *ACM Transactions on Graphics (Proc. SIGGRAPH) 21*, 3.

FATTAL, R., AGRAWALA, M., AND RUSINKIEWICZ, S. 2007. Multiscale shape and detail enhancement from multi-light image collections. *ACM Transactions on Graphics (Proc. SIGGRAPH) 26*, 3.

FATTAL, R., CARROLL, R., AND AGRAWALA, M. 2009. Edge-based image coarsening. *ACM Transactions on Graphics 29*, 1.

FATTAL, R. 2009. Edge-avoiding wavelets and their applications. *ACM Transactions on Graphics (Proc. SIGGRAPH) 28*, 3.

HE, K., SUN, J., AND TANG, X. 2010. Guided image filtering. In *Proceedings of European Conference on Computer Vision*.

HEEGER, D. J., AND BERGEN, J. R. 1995. Pyramid-based texture analysis/synthesis. In *Proceedings of the ACM SIGGRAPH conference*.

KASS, M., AND SOLOMON, J. 2010. Smoothed local histogram filters. *ACM Transactions on Graphics (Proc. SIGGRAPH) 29*, 3.

KIMMEL, R. 2003. *Numerical Geometry of Images: Theory, Algorithms, and Applications*. Springer. ISBN 0387955623.

LI, Y., SHARAN, L., AND ADELSON, E. H. 2005. Compressing and companding high dynamic range images with subband architectures. *ACM Transactions on Graphics (Proc. SIGGRAPH) 24*, 3.

LISCHINSKI, D., FARBMAN, Z., UYTTENDAELE, M., AND SZELISKI, R. 2006. Interactive local adjustment of tonal values. *ACM Transactions on Graphics (Proc. SIGGRAPH) 25*, 3.

MANTIUK, R., MYSZKOWSKI, K., AND SEIDEL, H.-P. 2006. A perceptual framework for contrast processing of high dynamic range images. *ACM Transactions on Applied Perception 3*, 3, 286–308.

MANTIUK, R., MANTIUK, R., TOMASZEWSKA, A., AND HEIDRICH, W. 2009. Color correction for tone mapping. *Computer Graphics Forum (Proc. Eurographics) 28*, 2, 193–202.

MASIA, B., AGUSTIN, S., FLEMING, R. W., SORKINE, O., AND GUTIERREZ, D. 2009. Evaluation of reverse tone mapping through varying exposure conditions. *ACM Transactions on Graphics (Proc. SIGGRAPH Asia) 28*, 5.

PARIS, S., AND DURAND, F. Tone-mapping code. http://people.csail.mit.edu/sparis/code/src/tone_mapping.zip. Accessed on January 14th, 2011.

PARIS, S., KORNPROBST, P., TUMBLIN, J., AND DURAND, F. 2009. Bilateral filtering: Theory and applications. *Foundations and Trends in Computer Graphics and Vision*.

PERONA, P., AND MALIK, J. 1990. Scale-space and edge detection using anisotropic diffusion. *IEEE Transactions Pattern Analysis Machine Intelligence 12*, 7, 629–639.

REINHARD, E., STARK, M., SHIRLEY, P., AND FERWERDA, J. 2002. Photographic tone reproduction for digital images. *ACM Transactions on Graphics (Proc. SIGGRAPH) 21*, 3.

SUBR, K., SOLER, C., AND DURAND, F. 2009. Edge-preserving multiscale image decomposition based on local extrema. *ACM Transactions on Graphics (Proc. SIGGRAPH Asia) 28*, 5.

SUNKAVALLI, K., JOHNSON, M. K., MATUSIK, W., AND PFISTER, H. 2010. Multi-scale image harmonization. *ACM Transactions on Graphics (Proc. SIGGRAPH) 29*, 3.

SZELISKI, R. 2006. Locally adapted hierarchical basis preconditioning. *ACM Transactions on Graphics (Proc. SIGGRAPH) 25*, 3.

TOMASI, C., AND MANDUCHI, R. 1998. Bilateral filtering for gray and color images. In *Proceedings of the International Conference on Computer Vision*, IEEE, 839–846.

TUMBLIN, J., AND TURK, G. 1999. Low curvature image simplifiers (LCIS): A boundary hierarchy for detail-preserving contrast reduction. In *Proceedings of SIGGRAPH*.

VUYLSTEKE, P., AND SCHOETERS, E. P. 1994. Multiscale image contrast amplification (MUSICA). In *Proceedings SPIE*, vol. 2167, 551–560.

WITKIN, A., TERZOPOULOS, D., AND KASS, M. 1987. Signal matching through scale space. *International Journal of Computer Vision 1*, 2, 759–764.

WITKIN, A. 1983. Scale-space filtering. In *Proceedings of the International Joint Conference on Artificial Intelligence*, vol. 2, 1019–1022.

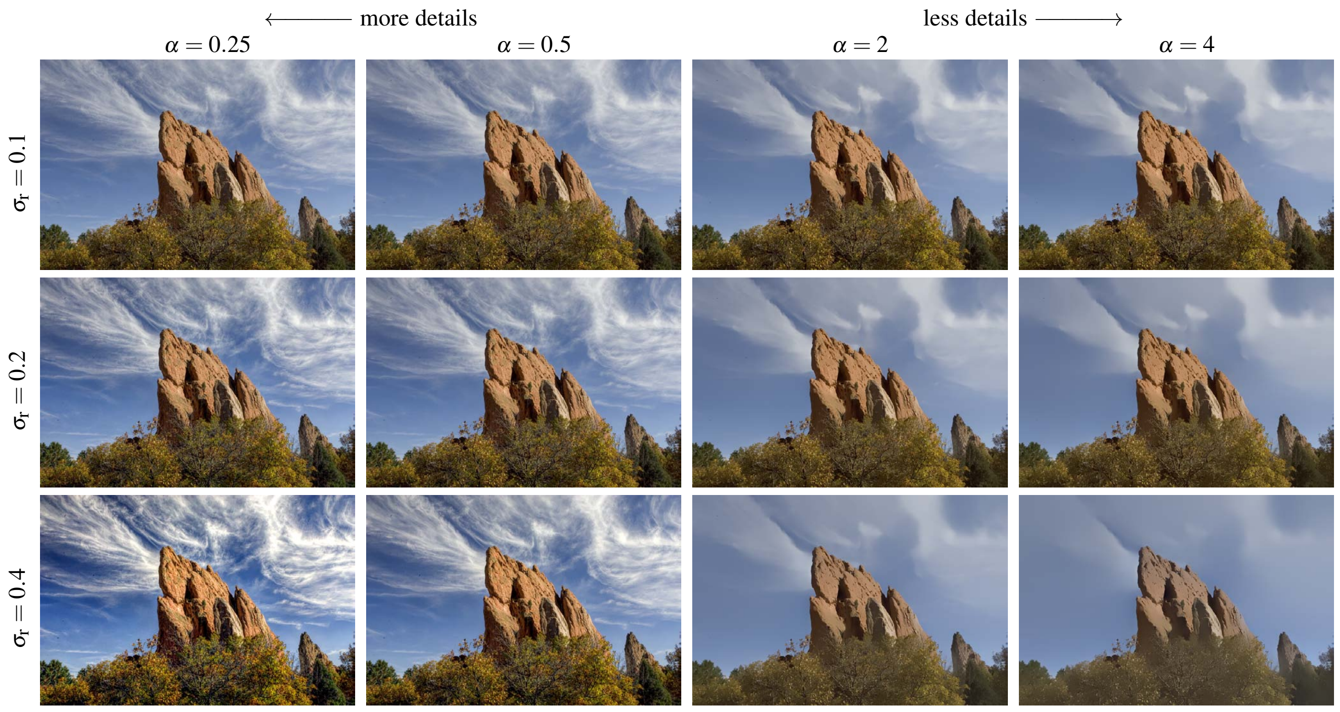


Figure 16: *Our filter to enhance and reduce details covers a large space of possible outputs without creating halos.*

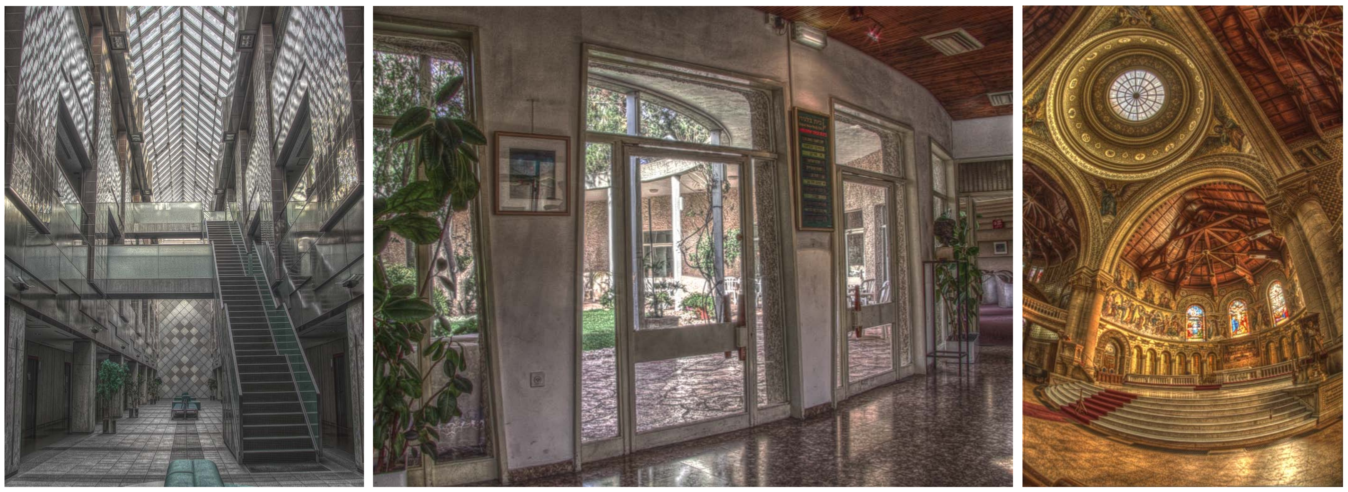


Figure 17: *We stressed our approach by applying a strong range compression coupled with a large detail increase ($\alpha = 0.25$, $\beta = 0$, $\sigma_r = \log(2.5)$). The results are dominated by local contrast and are reminiscent of the popular, exaggerated "HDR look" but without the unsightly halos associated with it. In terms of image quality, our results remain artifact-free in most cases. We explore further parameter variations in the supplemental material.*