Guided Image Filtering

Kaiming He[1], Jian Sun[2], and Xiaoou Tang[1,3]

[1] Department of Information Engineering, The Chinese University of Hong Kong
[2] Microsoft Research Asia
[3] Shenzhen Institutes of Advanced Technology, Chinese Academy of Sciences, China

Abstract. In this paper, we propose a novel type of explicit image filter - *guided filter*. Derived from a local linear model, the guided filter generates the filtering output by considering the content of a guidance image, which can be the input image itself or another different image. The guided filter can perform as an edge-preserving smoothing operator like the popular bilateral filter [1], but has better behavior near the edges. It also has a theoretical connection with the matting Laplacian matrix [2], so is a more generic concept than a smoothing operator and can better utilize the structures in the guidance image. Moreover, the guided filter has a fast and non-approximate linear-time algorithm, whose computational complexity is independent of the filtering kernel size. We demonstrate that the guided filter is both effective and efficient in a great variety of computer vision and computer graphics applications including noise reduction, detail smoothing/enhancement, HDR compression, image matting/feathering, haze removal, and joint upsampling.

1 Introduction

Most applications in computer vision and computer graphics involve the concept of image filtering to reduce noise and/or extract useful image structures. Simple explicit linear translation-invariant (LTI) filters like Gaussian filter, Laplacian filter, and Sobel filter are widely used in image blurring/sharpening, edge detection, and feature extraction [3]. LTI filtering also includes the process of solving a Poisson Equation, such as in high dynamic range (HDR) compression [4], image stitching [5], and image matting [6], where the filtering kernel is implicitly defined by the inverse of a homogenous Laplacian matrix.

The kernels of LTI filters are spatially invariant and independent of any image content. But in many cases, we may want to incorporate additional information from a given *guidance* image during the filtering process. For example, in colorization [7] the output chrominance channels should have consistent edges with the given luminance channel; in image matting [2] the output alpha matte should capture the thin structures like hair in the image. One approach to achieve this purpose is to optimize a quadratic function that directly enforces some constraints on the unknown output by considering the guidance image. The solution is then obtained by solving a large sparse matrix encoded with the information of the guidance image. This inhomogeneous matrix implicitly defines a *translation-variant* filtering kernel. This approach is widely used in many

applications, like colorization [7], image matting [2], multi-scale decomposition [8], and haze removal [9]. While this optimization-based approach often yields the state-of-the-art quality, it comes with the price of long computational time.

The other approach is to explicitly build the filter kernels using the guidance image. The bilateral filter, proposed in [10], made popular in [1], and later generalized in [11], is perhaps the most popular one of such filters. Its output at a pixel is a weighted average of the nearby pixels, where the weights depend on the intensity/color similarities in the guidance image. The guidance image can be the filter input itself [1] or another image [11]. The bilateral filter can smooth small fluctuations and preserve edges. While this filter is effective in many situations, it may have unwanted gradient reversal artifacts [12, 13, 8] near edges (further explained in Section 3.4). Its fast implementation is also a challenging problem. Recent techniques [14–17] rely on quantization methods to accelerate but may sacrifice the accuracy.

In this paper we propose a new type of explicit image filter, called *guided filter*. The filtering output is locally a linear transform of the guidance image. This filter has the edge-preserving smoothing property like the bilateral filter, but does not suffer from the gradient reversal artifacts. It is also related to the matting Laplacian matrix [2], so is a more generic concept and is applicable in other applications beyond the scope of "smoothing". Moreover, the guided filter has an O(N) time (in the number of pixels N) *exact* algorithm for both gray-scale and color images. Experiments show that the guided filter performs very well in terms of both quality and efficiency in a great variety of applications, such as noise reduction, detail smoothing/enhancement, HDR compression, image matting/feathering, haze removal, and joint upsampling.

2 Related Work

2.1 Bilateral Filter

The bilateral filter computes the filter output at a pixel as a weighted average of neighboring pixels. It smoothes the image while preserving edges. Due to this nice property, it has been widely used in noise reduction [18], HDR compression [12], multi-scale detail decomposition [19], and image abstraction [20]. It is generalized to the joint bilateral filter in [11], in which the weights are computed from another guidance image rather than the filter input. The joint bilateral filter is particular favored when the filter input is not reliable to provide edge information, e.g., when it is very noisy or is an intermediate result. The joint bilateral filter is applicable in flash/no-flash denoising [11], image upsamling [21], and image deconvolution [22].

However, it has been noticed [12, 13, 8] that the bilateral filter may have the gradient reversal artifacts in detail decomposition and HDR compression. The reason is that when a pixel (often on an edge) has few similar pixels around it, the Gaussian weighted average is unstable. Another issue concerning the bilateral filter is its efficiency. The brute-force implementation is in $O(Nr^2)$

time, which is prohibitively high when the kernel radius r is large. In [14] an approximated solution is obtained in a discretized space-color grid. Recently, $\mathrm{O}(N)$ time algorithms [15, 16] have been developed based on histograms. Adams et al. [17] propose a fast algorithm for color images. All the above methods require a high quantization degree to achieve satisfactory speed, but at the expense of quality degradation.

2.2 Optimization-based Image Filtering

A series of approaches optimize a quadratic cost function and solve a linear system, which is equivalent to implicitly filtering an image by an inverse matrix. In image segmentation [23] and colorization [7], the affinities of this matrix are Gaussian functions of the color similarities. In image matting, a matting Laplacian matrix [2] is designed to enforce the alpha matte as a local linear transform of the image colors. This matrix is also applicable in haze removal [9]. The weighted least squares (WLS) filter in [8] adjusts the matrix affinities according to the image gradients and produces a halo-free decomposition of the input image. Although these optimization-based approaches often generate high quality results, solving the corresponding linear system is time-consuming.

It has been found that the optimization-based filters are closely related to the explicit filters. In [24] Elad shows that the bilateral filter is one Jacobi iteration in solving the Gaussian affinity matrix. In [25] Fattal defines the edge-avoiding wavelets to approximate the WLS filter. These explicit filters are often simpler and faster than the optimization-based filters.

3 Guided Filter

We first define a general linear translation-variant filtering process, which involves a guidance image I, an input image p, and an output image q. Both I and p are given beforehand according to the application, and they can be identical. The filtering output at a pixel i is expressed as a weighted average:

$$q_i = \sum_j W_{ij}(I) p_j, \tag{1}$$

where i and j are pixel indexes. The filter kernel W_{ij} is a function of the guidance image I and independent of p. This filter is linear with respect to p.

A concrete example of such a filter is the joint bilateral filter [11]. The bilateral filtering kernel W^{bf} is given by:

$$W^{\mathrm{bf}}_{ij}(I) = \frac{1}{K_i} \exp(-\frac{|\mathbf{x}_i - \mathbf{x}_j|^2}{\sigma_{\mathrm{s}}^2}) \exp(-\frac{|I_i - I_j|^2}{\sigma_{\mathrm{r}}^2}). \tag{2}$$

where $\mathbf{x}$ is the pixel coordinate, and K_i is a normalizing parameter to ensure that $\sum_j W^{\mathrm{bf}}_{ij} = 1$. The parameters σ_{s} and σ_{r} adjust the spatial similarity and the range (intensity/color) similarity respectively. The joint bilateral filter degrades to the original bilateral filter [1] when I and p are identical.

3.1 Definition

Now we define the guided filter and its kernel. The key assumption of the guided filter is a local linear model between the guidance I and the filter output q. We assume that q is a linear transform of I in a window ω_k centered at the pixel k:

$$q_i = a_k I_i + b_k, \forall i \in \omega_k, \tag{3}$$

where (a_k, b_k) are some linear coefficients assumed to be constant in ω_k. We use a square window of a radius r. This local linear model ensures that q has an edge only if I has an edge, because $\nabla q = a \nabla I$. This model has been proven useful in image matting [2], image super-resolution [26], and haze removal [9].

To determine the linear coefficients, we seek a solution to (3) that minimizes the difference between q and the filter input p. Specifically, we minimize the following cost function in the window:

$$E(a_k, b_k) = \sum_{i \in \omega_k} ((a_k I_i + b_k - p_i)^2 + \epsilon a_k^2). \tag{4}$$

Here ϵ is a regularization parameter preventing a_k from being too large. We will investigate its significance in Section 3.2. The solution to (4) can be given by linear regression [27]:

$$a_k = \frac{\frac{1}{|\omega|} \sum_{i \in \omega_k} I_i p_i - \mu_k \bar{p}_k}{\sigma_k^2 + \epsilon} \tag{5}$$

$$b_k = \bar{p}_k - a_k \mu_k. \tag{6}$$

Here, μ_k and σ_k^2 are the mean and variance of I in ω_k, $|\omega|$ is the number of pixels in ω_k, and $\bar{p}_k = \frac{1}{|\omega|} \sum_{i \in \omega_k} p_i$ is the mean of p in ω_k.

Next we apply the linear model to all local windows in the entire image. However, a pixel i is involved in all the windows ω_k that contain i, so the value of q_i in (3) is not the same when it is computed in different windows. A simple strategy is to average all the possible values of q_i. So after computing (a_k, b_k) for all patches ω_k in the image, we compute the filter output by:

$$q_i = \frac{1}{|\omega|} \sum_{k: i \in \omega_k} (a_k I_i + b_k) \tag{7}$$

$$= \bar{a}_i I_i + \bar{b}_i \tag{8}$$

where $\bar{a}_i = \frac{1}{|\omega|} \sum_{k \in \omega_i} a_k$ and $\bar{b}_i = \frac{1}{|\omega|} \sum_{k \in \omega_i} b_k$.

With this modification ∇q is no longer scaling of ∇I, because the linear coefficients $(\bar{a}_i, \bar{b}_i)$ vary spatially. But since $(\bar{a}_i, \bar{b}_i)$ are the output of an average filter, their gradients should be much smaller than that of I near strong edges. In this situation we can still have $\nabla q \approx \bar{a} \nabla I$, meaning that abrupt intensity changes in I can be mostly maintained in q.

We point out that the relationship among I, p, and q given by (5), (6), and (8) are indeed in the form of image filtering (1). In fact, a_k in (5) can be

rewritten as a weighted sum of p: $a_k = \sum_j A_{kj}(I)p_j$, where A_{ij} are the weights only dependent on I. For the same reason, we also have $b_k = \sum_j B_{kj}(I)p_j$ from (6) and $q_i = \sum_j W_{ij}(I)p_j$ from (8). It can be proven (see the supplementary materials) that the kernel weights can be explicitly expressed by:

$$W_{ij}(I) = \frac{1}{|\omega|^2} \sum_{k:(i,j)\in\omega_k} (1 + \frac{(I_i - \mu_k)(I_j - \mu_k)}{\sigma_k^2 + \epsilon}). \tag{9}$$

Some further computations show that $\sum_j W_{ij}(I) = 1$. No extra effort is needed to normalize the weights.

3.2 Edge-preserving Filtering

Fig. 1 (top) shows an example of the guided filter with various sets of parameters. We can see that it has the edge-preserving smoothing property. This can be explained intuitively as following. Consider the case that $I = p$. It is clear that if $\epsilon = 0$, then the solution to (4) is $a_k = 1$ and $b_k = 0$. If $\epsilon > 0$, we can consider two cases:

Case 1: "Flat patch". If the image I is constant in ω_k, then (4) is solved by $a_k = 0$ and $b_k = \bar{p}_k$;

Case 2: "High variance". If the image I changes a lot within ω_k, then a_k becomes close to 1 while b_k is close to 0.

When a_k and b_k are averaged to get $\bar{a}_i$ and $\bar{b}_i$, combined in (8) to get the output, we have that if a pixel is in the middle of a "high variance" area, then its value is unchanged, whereas if it is in the middle of a "flat patch" area, its value becomes the average of the pixels nearby.

More specifically, the criterion of a "flat patch" or a "high variance" is given by the parameter ϵ. The patches with variance (σ^2) much smaller than ϵ are smoothed, whereas those with variance much larger than ϵ are preserved. The effect of ϵ in the guided filter is similar with the range variance σ_{r}^2 in the bilateral filter (2). Both parameters determine "what is an edge/a high variance patch that should be preserved". Fig. 1 (bottom) shows the bilateral filter results as a comparison.

3.3 Filter Kernel

The edge-preserving smoothing property can also be understood by investigating the filter kernel (9). Take an ideal step edge of a 1-D signal as an example (Fig. 2). The terms $I_i - \mu_k$ and $I_j - \mu_k$ have the same sign (+/-) when I_i and I_j are on the same side of an edge, while they have opposite signs when the two pixels are on different sides. So in (9) the term $1 + \frac{(I_i-\mu_k)(I_j-\mu_k)}{\sigma_k^2+\epsilon}$ is much smaller (and close to zero) for two pixels on different sides than on the same sides. This means that the pixels across an edge are almost not averaged together. We can also understand the smoothing effect of ϵ from (9). When $\sigma_k^2 \ll \epsilon$ ("flat patch"), the kernel becomes $W_{ij}(I) = \frac{1}{|\omega|^2} \sum_{k:(i,j)\in\omega_k} 1$: this is a low-pass filter that biases neither side of an edge.

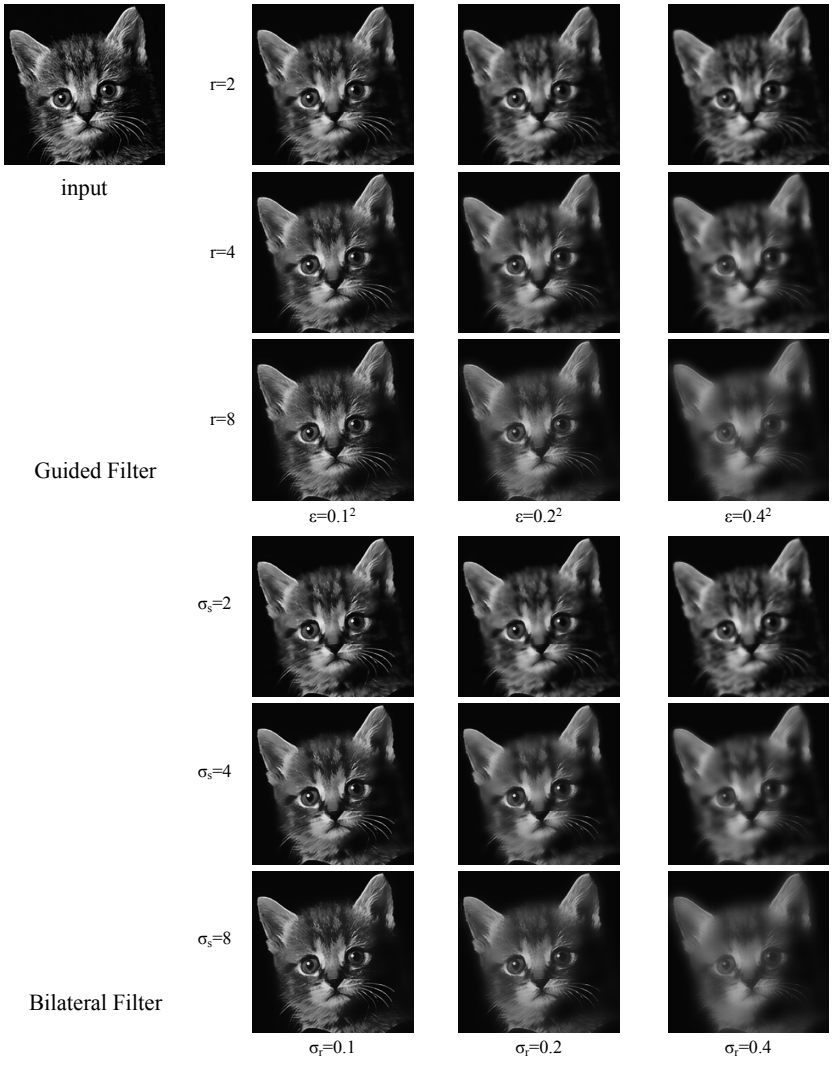

Fig. 1. The filtered images of a gray-scale input. In this example the guidance I is identical to the input p. The input image has intensity in [0, 1]. The input image is from [1].

Fig. 3 shows two examples of the kernel shapes in real images. In the top row are the kernels near a step edge. Like the bilateral kernel, the guided filter's kernel assigns nearly zero weights to the pixels on the opposite side of the edge. In the bottom row are the kernels in a patch with small scale textures. Both filters average almost all the nearby pixels together and appear as low-pass filters.

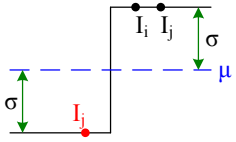

Fig. 2. A 1-D example of an ideal step edge. For a window that exactly center on the edge, the variables μ and σ are as indicated.

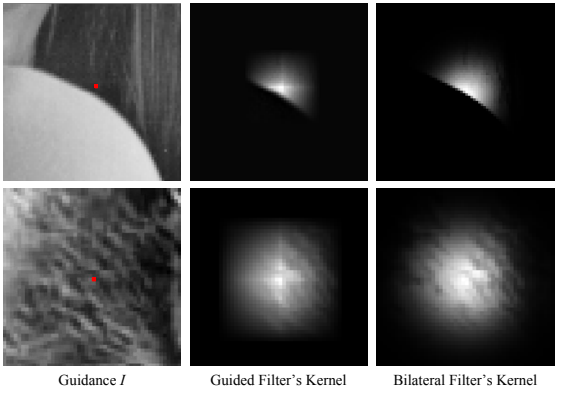

Fig. 3. Filter kernels. Top: a step edge (guided filter: $r = 7, \epsilon = 0.1^2$, bilateral filter: $\sigma_s = 7, \sigma_r = 0.1$). Bottom: a textured patch (guided filter: $r = 8, \epsilon = 0.2^2$, bilateral filter: $\sigma_s = 8, \sigma_r = 0.2$). The kernels are centered at the pixels denote by the red dots.

3.4 Gradient Preserving Filtering

Though the guided filter is an edge-preserving smoothing filter like the bilateral filter, it avoids the gradient reversal artifacts that may appear in detail enhancement and HDR compression. Fig. 4 shows a 1-D example of detail enhancement. Given the input signal (black), its edge-preserving smoothed output is used as a *base layer* (red). The difference between the input signal and the base layer is the *detail layer* (blue). It is magnified to boost the details. The enhanced signal (green) is the combination of the boosted detail layer and the base layer. An elaborate description of this method can be found in [12].

For the bilateral filter (Fig. 4 left), the base layer is not consistent with input signal at the edge pixels. This is because few pixels around them have similar colors, and the Gaussian weighted average has little statistical data and becomes unreliable. So the detail layer has great fluctuations, and the recombined signal has reversed gradients as shown in the figure. On the other hand, the guided filter (Fig. 4 right) better preserves the gradient information in I, because the gradient of the base layer is $\nabla q \approx \bar{a} \nabla I$ near the edge. The shape of the edge is well maintained in the recombined layer.

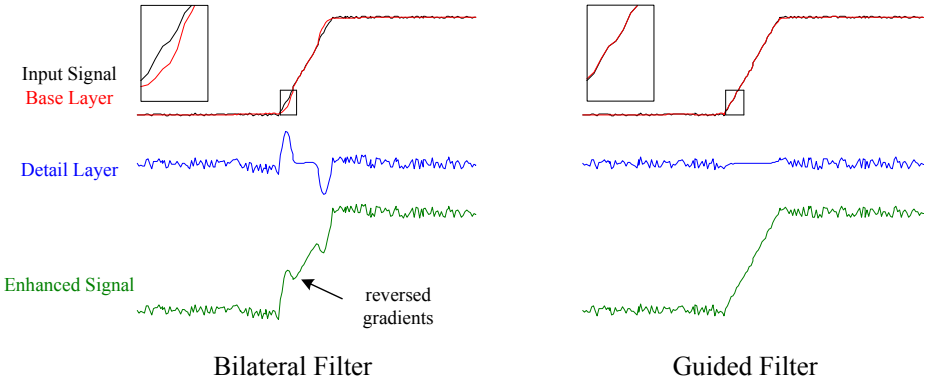

Fig. 4. 1-D illustration for detail enhancement. See the text for explanation.

3.5 Relation to the Matting Laplacian Matrix

The guided filter can not only be used as a smoothing operator. It is also closely related to the matting Laplacian matrix [2]. This casts new insights into the guided filter and inspires some new applications.

In a closed-form solution to matting [2], the matting Laplacian matrix is derived from a local linear model. Unlike the guided filter which computes the local optimal for each window, the closed-form solution seeks a global optimal. To solve for the unknown alpha matte, this method minimizes the following cost function:

$$E(\alpha) = (\alpha - \beta)^{\mathrm{T}} \Lambda (\alpha - \beta) + \alpha^{\mathrm{T}} \mathrm{L} \alpha, \tag{10}$$

where α is the unknown alpha matte denoted in its matrix form, β is the constraint (e.g., a trimap), L is an N×N matting Laplacian matrix, and Λ is a diagonal matrix encoded with the weights of the constraints. The solution to this optimization problem is given by solving a linear system: $(\mathrm{L} + \Lambda)\alpha = \Lambda\beta$.

The elements of the matting Laplacian matrix are given by:

$$\mathrm{L}_{ij} = \sum_{k:(i,j)\in\omega_k} (\delta_{ij} - \frac{1}{|\omega|}(1 + \frac{(I_i - \mu_k)(I_j - \mu_k)}{\sigma_k^2 + \epsilon})). \tag{11}$$

where δ_{ij} is the Kronecker delta. Comparing (11) with (9), we find that the elements of the matting Laplacian matrix can be directly given by the guided filter kernel weights:

$$\mathrm{L}_{ij} = |\omega|(\delta_{ij} - \mathrm{W}_{ij}), \tag{12}$$

Following the strategy in [24], we can further prove (see the supplementary materials) that the output of the guided filter is one Jacobi iteration in optimizing (10). If β is a reasonably good guess of the matte, we can run one Jacobi step and obtain an approximate solution to (10) by a guided filtering process: $\alpha_i \approx \sum_j W_{ij}(I)\beta_j$. In Section 4, we apply this property to image matting/feathering and haze removal.

3.6 O(N) Time Exact Algorithm

One more advantage of the guided filter over the bilateral filter is that it automatically has an O(N) time exact algorithm. O(N) time implies that the time complexity is independent of the window radius r, so we are free to use arbitrary kernel sizes in the applications.

The filtering process in (1) is a translation-variant convolution. Its computational complexity increases when the kernel becomes larger. Instead of directly performing the convolution, we compute the filter output from its definition (5)(6)(8). All the summations in these equations are box filters ($\sum_{i\in\omega_k} f_i$). We apply the O(N) time Integral Image technique [28] to calculate the output of a box filter. So the guided filter can be computed in O(N) time.

The O(N) time algorithm can be easily extended to RGB color guidance images. Filtering using color guidance images is necessary when the edges or details are not discriminable in any single channel. To generalize to a color guidance image, we rewrite the local linear model (3) as:

$$q_i = \mathbf{a}_k^{\mathrm{T}}\mathbf{I}_i + b_k, \forall i \in \omega_k. \tag{13}$$

Here $\mathbf{I}_i$ is a 3×1 color vector, $\mathbf{a}_k$ is a 3×1 coefficient vector, q_i and b_k are scalars. The guided filter for color guidance images becomes:

$$\mathbf{a}_k = (\Sigma_k + \epsilon \mathrm{U})^{-1}(\frac{1}{|\omega|}\sum_{i\in\omega_k}\mathbf{I}_i p_i - \mu_k \bar{p}_k) \tag{14}$$

$$b_k = \bar{p}_k - \mathbf{a}_k^{\mathrm{T}}\mu_k \tag{15}$$

$$q_i = \bar{\mathbf{a}}_i^{\mathrm{T}}\mathbf{I}_i + \bar{b}_i. \tag{16}$$

Here Σ_k is the 3×3 covariance matrix of $\mathbf{I}$ in ω_k, and U is a 3×3 identity matrix. The summations are still box filters and can be computed in O(N) time.

We experiment the running time in a laptop with a 2.0Hz Intel Core 2 Duo CPU. For the gray-scale guided filter, it takes 80ms to process a 1-megapixel image. As a comparison, the O(N) time bilateral filter in [15] requires 42ms using a histogram of 32 bins, and 85ms using 64 bins. Note that the guided filter algorithm is non-approximate and applicable for data of high bit-depth, while the O(N) time bilateral filter may have noticeable quantization artifacts (see Fig. 5). The algorithm in [16] requires 1.2 seconds per megapixel using 8 bins (using the public code on the authors' website). For RGB guidance images, the guided filter takes about 0.3s to process a 1-megapixel image. The algorithm for high-dimensional bilateral filter in [16] takes about 10 seconds on average to process per 1-megapixel RGB image.

4 Applications and Experimental Results

In this section, we apply the guided filter to a great variety of computer vision and graphics applications.

Fig. 5. Quantization artifacts of O(N) time bilateral filter. (a) Input HDR image (32bit float, displayed by linear scaling). (b) Compressed image using the O(N) bilateral filter in [15] (64 bins). (c) Compressed image using the guided filter. This figure is best viewed in the electronic version of this paper.

Detail Enhancement and HDR Compression. The method for detail enhancement is described in Section 3.4. For HDR compression, we compress the base layer instead of magnifying the detail layer. Fig. 6 shows an example for detail enhancement, and Fig. 7 shows an example for HDR Compression. The results using the bilateral filter are also provided. As shown in the zoom-in patches, the bilateral filter leads to gradient reversal artifacts.

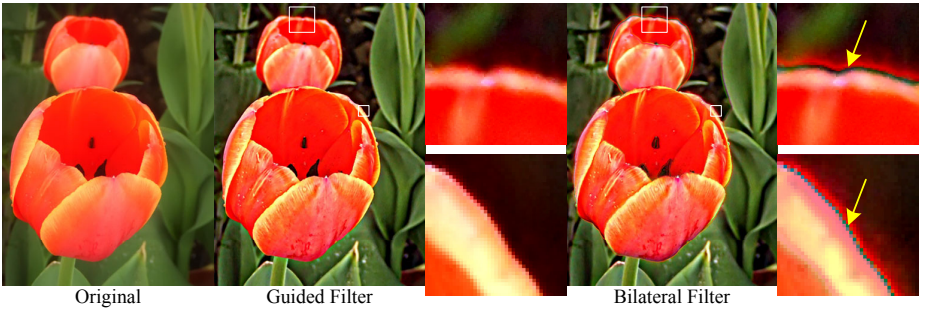

Fig. 6. Detail enhancement. The parameters are $r = 16$, $\epsilon = 0.1^2$ for the guided filter, and $\sigma_s = 16$, $\sigma_r = 0.1$ for the bilateral filter. The detail layer is boosted $\times 5$.

Flash/No-flash Denoising. In [11] it is proposed to denoise a no-flash image under the guidance of its flash version. Fig. 8 shows a comparison of using the joint bilateral filter and the guided filter. The gradient reversal artifacts are noticeable near some edges in the joint bilateral filter result.

Matting/Guided Feathering. We apply the guided filter as *guided feathering*: a binary mask is refined to appear an alpha matte near the object boundaries

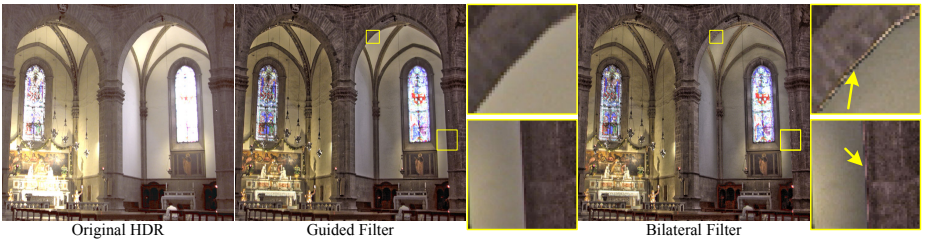

Fig. 7. HDR compression. The parameters are $r = 15$, $\epsilon = 0.12^2$ for the guided filter, and $\sigma_s = 15$, $\sigma_r = 0.12$ for the bilateral filter.

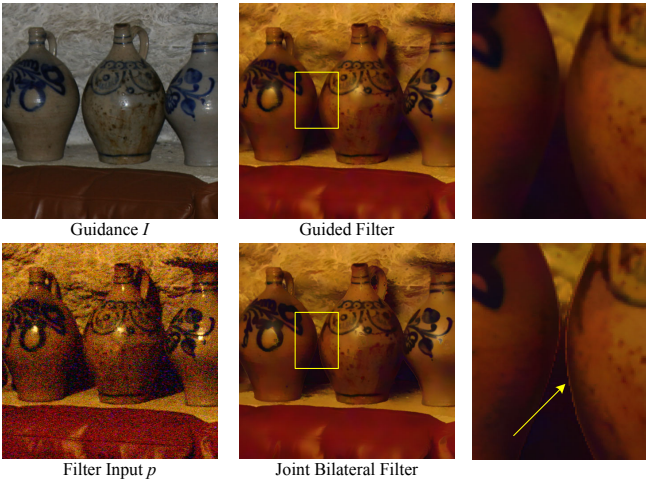

Fig. 8. Flash/no-flash denoising. The parameters are $r = 8$, $\epsilon = 0.2^2$ for the guided filter, and $\sigma_s = 8$, $\sigma_r = 0.2$ for the joint bilateral filter.

(Fig. 9). The binary mask can be obtained from graph-cut or other segmentation methods, and is used as the filter input p. The guidance $\mathbf{I}$ is the color image. A similar function "Refine Edge" can be found in the commercial software Adobe Photoshop CS4. We can also compute an accurate matte using the closed-form solution [2]. In Fig. 9 we compare our results with the Photoshop Refine Edge and the closed-form solution. Our result is visually comparable with the closed-form solution in this short hair case. Both our method and Photoshop provide fast feedback (<1s) for this 6-mega-pixel image, while the closed-form solution takes about two minutes to solve a huge linear system.

Single Image Haze Removal. In [9] a haze transmission map is roughly estimated using a dark channel prior, and is refined by solving the matting Laplacian matrix. On the contrary, we simply filter the raw transmission map under the guidance of the hazy image. The results are visually similar (Fig. 10). The guid-

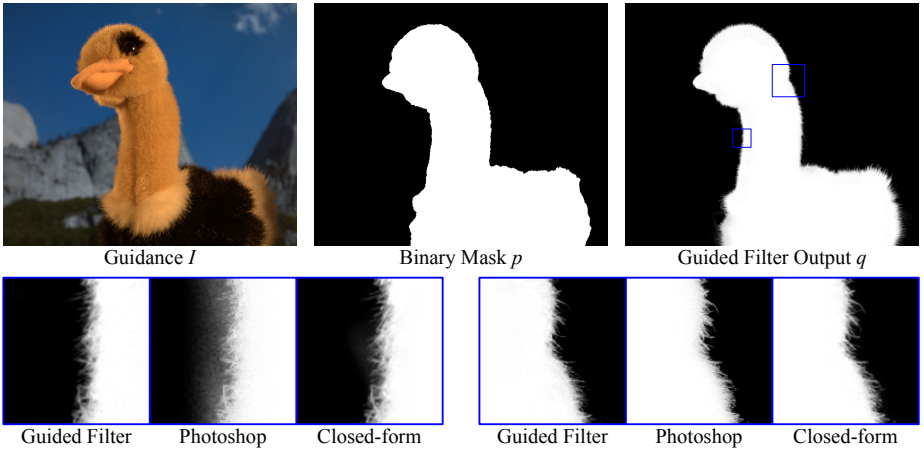

Fig. 9. Guided Feathering. A binary mask p is filtered under the guidance of I. In the zoom-in patches, we compare with the Photoshop Refine Edge function and the closed-form matting. For closed-form matting, we erode and dilate the mask to obtain a trimap. The parameters are $r = 60$, $\epsilon = 10^{-6}$ for the guided filter.

ed filter takes about 0.1s to process this 600×400 color image, but the running time is over 10 seconds as reported in [9] .

Joint Upsampling. Joint upsampling [21] is to upsample an image under the guidance of another image. Taking the application of colorization [7] as an example. A gray-scale luminance image is colorized through an optimization process. To reduce the running time, the chrominance channels are solved at a coarse resolution and upsampled under the guidance of the full resolution luminance image by the joint bilateral filter [21]. This upsampling process can also be performed by the guided filter. The result is visually comparable (Fig. 11).

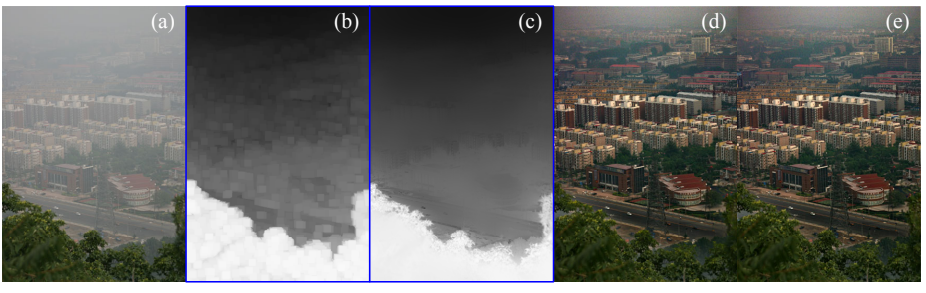

Fig. 10. Haze Removal. (a) Hazy image. (b) Raw transmission map [9]. (c) The raw transmission map is refined by the guided filter ($r = 20$, $\epsilon = 10^{-3}$). (e) Recovered image using (c). (d) The result in [9].

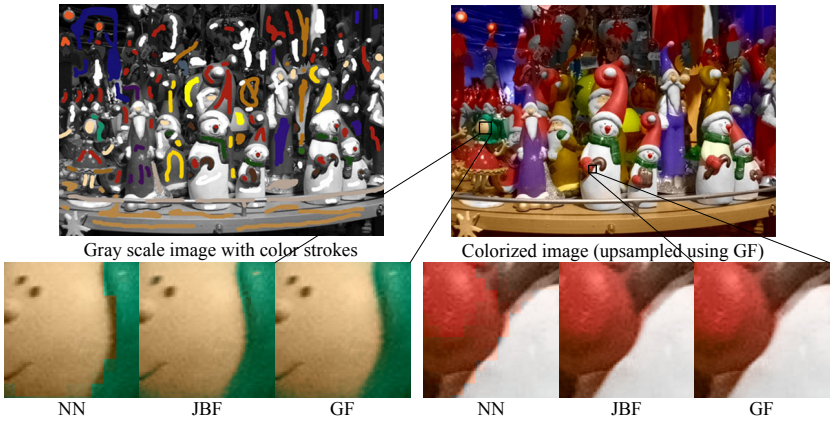

Fig. 11. Joint Upsampling for Colorization. The upsampling methods includes: nearest-neighbor (NN), joint bilateral filter (JBF), and guided filter (GF).

5 Discussion and Conclusion

In this paper, we have presented a novel filter which is widely applicable in computer vision and graphics. Different from the recent trend towards accelerating the bilateral filter [14–17], we define a new type of filter that shares the nice property of edge-preserving smoothing but can be computed efficiently and exactly. Our filter is more generic and can handle some applications beyond the concept of "smoothing". Since the local linear model (3) can be regarded as a simple case of learning, other advanced models/features might be applied to obtain new filters.

As a locally based operator, the guided filter is not directly applicable for sparse inputs like strokes. It also shares a common limitation of other explicit filter - it may have halos near some edges. In fact, it is ambiguous for a low-level and local operator to determine which edge should be smoothed and which should be preserved. Unsuitably smoothing an edge will result in halos near it. However, we believe that the simplicity and efficiency of the guided filter still make it beneficial in many situations.

References

1. Tomasi, C., Manduchi, R.: Bilateral filtering for gray and color images. ICCV (1998)
2. Levin, A., Lischinski, D., Weiss, Y.: A closed form solution to natural image matting. CVPR (2006)
3. Gonzalez, R.C., Woods, R.E.: Digital Image Processing. 2 edn. Prentice Hall (2002)

4. Fattal, R., Lischinski, D., Werman, M.: Gradient domain high dynamic range compression. SIGGRAPH (2002)
5. Pérez, P.: Poisson image editing. SIGGRAPH (2003)
6. Sun, J., Jia, J., Tang, C.K., Shum, H.Y.: Poisson matting. SIGGRAPH (2004)
7. Levin, A., Lischinski, D., Weiss, Y.: Colorization using optimization. SIGGRAPH (2004)
8. Farbman, Z., Fattal, R., Lischinski, D., Szeliski, R.: Edge-preserving decompositions for multi-scale tone and detail manipulation. SIGGRAPH (2008)
9. He, K., Sun, J., Tang, X.: Single image haze removal using dark channel prior. CVPR (2009)
10. Aurich, V., Weule, J.: Non-linear gaussian filters performing edge preserving diffusion. In: Mustererkennung 1995, 17. DAGM-Symposium, Springer-Verlag (1995) 538–545
11. Petschnigg, G., Agrawala, M., Hoppe, H., Szeliski, R., Cohen, M., Toyama, K.: Digital photography with flash and no-flash image pairs. SIGGRAPH (2004)
12. Durand, F., Dorsey, J.: Fast bilateral filtering for the display of high-dynamic-range images. SIGGRAPH (2002)
13. Bae, S., Paris, S., Durand, F.: Two-scale tone management for photographic look. SIGGRAPH (2006)
14. Paris, S., Durand, F.: A fast approximation of the bilateral filter using a signal processing approach. ECCV (2006)
15. Porikli, F.: Constant time o(1) bilateral filtering. CVPR (2008)
16. Yang, Q., Tan, K.H., Ahuja, N.: Real-time o(1) bilateral filtering. CVPR (2009)
17. Adams, A., Gelfand, N., Dolson, J., Levoy, M.: Gaussian kd-trees for fast high-dimensional filtering. SIGGRAPH (2009)
18. Liu, C., Freeman, W.T., Szeliski, R., Kang, S.B.: Noise estimation from a single image. CVPR (2006)
19. Fattal, R., Agrawala, M., Rusinkiewicz, S.: Multiscale shape and detail enhancement from multi-light image collections. SIGGRAPH (2007)
20. Winnemöller, H., Olsen, S.C., Gooch, B.: Real-time video abstraction. SIGGRAPH (2006)
21. Kopf, J., Cohen, M., Lischinski, D., Uyttendaele, M.: Joint bilateral upsampling. SIGGRAPH (2007)
22. Yuan, L., Sun, J., Quan, L., Shum, H.Y.: Progressive inter-scale and intra-scale non-blind image deconvolution. SIGGRAPH (2008)
23. Weiss, Y.: Segmentation using eigenvectors: A unifying view. ICCV (1999)
24. Elad, M.: On the origin of the bilateral filter and ways to improve it. IEEE Transactions on Image Processing (2002)
25. Fattal, R.: Edge-avoiding wavelets and their applications. SIGGRAPH (2009)
26. Zomet, A., Peleg, S.: Multi-sensor super resolution. IEEE Workshop on Applications of Computer Vision (2002)
27. Draper, N., Smith, H.: Applied Regression Analysis. 2 edn. John Wiley (1981)
28. Crow, F.: Summed-area tables for texture mapping. SIGGRAPH (1984)