Propagated Image Filtering

Jen-Hao Rick Chang
Dept. Electrical and Computer Engineering
Carnegie Mellon University
rickchang@cmu.edu

Yu-Chiang Frank Wang
Research Center for IT Innovation
Academia Sinica
ycwang@citi.sinica.edu.tw

Abstract

We propose the propagation filter as a novel image filtering operator, with the goal of smoothing over neighboring image pixels while preserving image context like edges or textural regions. In particular, our filter does not to utilize explicit spatial kernel functions as bilateral and guided filters do. We will show that our propagation filter can be viewed as a robust estimator, which minimizes the expected difference between the filtered and desirable image outputs. We will also relate propagation filtering to belief propagation, and suggest techniques if further speedup of the filtering process is necessary. In our experiments, we apply our propagation filter to a variety of applications such as image denoising, smoothing, fusion, and high-dynamic-range (HDR) compression. We will show that improved performance over existing image filters can be achieved.

1. Introduction

Image filtering is a process of updating pixel values in an image to achieve particular goals like denoising, smoothing, enhancement, or matting. It typically requires the extraction of particular image characteristics, while undesirable patterns like noise or irrelevant textural regions need to be disregarded. If *cross-region mixing* occurs during the filtering process, i.e., the characteristics of adjacent image regions are blended, the output image would contain blurry regions which result in degraded visual quality.

Bilateral [22, 24] and guided filters [14] are popular edge-preserving image filters, which are able to alleviate the aforementioned problem during the filtering process. The idea of these filters is to observe and process neighboring pixels with similar pixel values, so that desirable image context can be preserved. Both bilateral and guided filters have been successfully applied to a variety of applications such as noise reduction [1, 16, 20, 8], tone management [2, 7, 21, 9], and image fusion [15]. However, these filters require predefined pixel neighborhood regions (via spatial functions or kernels), which are typically difficult to determine beforehand. For example (and as discussed later in Section 3), choosing a large neighborhood for highly textural regions would result in cross-region mixing, while selecting a small one would limit the filtering performance. Recently, Lu *et al*. [17] tackled this problem by thresholding neighborhoods within a predefined pixel value difference. Instead of predefined pixel neighborhoods, geodesic filtering [5, 12] dynamically determines their filtering kernels by calculating the accumulated difference between the values of adjacent pixels. As discussed in Section 3, although geodesic filters have shown promising results for denoising tasks, they still suffer from cross-region mixing in smoothing and other filtering tasks.

In this paper, we propose a *propagation filter* for solving the above tasks. Our propagation filter is able to observe and preserve image characteristics without the need to apply explicit spatial kernel functions. We will show that our filtering process can be regarded as a one-step estimator, which minimizes the expected error between the filtered and desirable image outputs. In the experiments, we consider several image processing applications such as image denoising/smoothing, image fusion, and high-dynamic-range (HDR) imaging. We will verify that our proposed filter would perform favorably against existing bilateral, guided, and geodesic filters on the above tasks.

2. Related Work

2.1. Bilateral Filtering

Well known as an edge-preserving filter, the bilateral filter [22] calculates the value for pixel s as follows:

$$I'_s = \frac{1}{Z_s} \sum_{t \in \Omega} g(d_{\text{BF}}(s,t);\sigma_s)\, g(d_{\text{BF}}(I_s, I_t);\sigma_r)\, I_t, \qquad (1)$$

where I'_s is the filtered output at pixel s, Ω denotes the set of pixels t in the input I, and $g(x;\sigma)$ is a Gaussian function with variance σ^2. For bilateral filters, the *spatial* and *photometric* distances between pixels s and t are defined as:

$$d_{\text{BF}}(s,t) = \|t - s\|, \text{ and } d_{\text{BF}}(I_s, I_t) = \|I_t - I_s\|, \qquad (2)$$

978-1-4673-6964-0/15/$31.00 ©2015 IEEE

which calculate the Euclidean distance between their locations and that between their pixel values, respectively. In (1), Z_s is the normalization term, which is calculated as $Z_s = \sum_{t\in\Omega} g(d_{\text{BF}}(s,t);\sigma_s)\, g(d_{\text{BF}}(I_s,I_t);\sigma_r)$. Thus, bilateral filtering considers both spatial and photometric distances between pixel values, and applies the weighted pixel value as the filtered output.

Yang [24] proposed a recursive version of bilateral filtering, which defines the photometric distance in (1) as

$$d_{\text{BF}}(I_s,I_t) = \sqrt{\sum_{x,\, x+1\in\phi} \|I_{x+1} - I_x\|^2}, \qquad (3)$$

where ϕ is a predefined path connecting pixels s and t. It can be seen that, recursive bilateral filtering calculates photometric distances by accumulating the distances between adjacent pixels along the path ϕ. While this modification decreases the computation costs for bilateral filtering, the price to pay is the ability in preserving image edges. In other words, cross-region mixing is more likely to occur.

2.2. Guided Filtering

The goal of guided filtering [14] is to filter the input image I based on a guidance image I^g. It assumes that the filtered output at pixel s is a linear transformation of the pixels (within a window W_k) of I^g. More specifically, the output image can be represented as $I'_s = \sum_{t\in\mathcal{N}(s)} w^g_{s,t}\, I_t$, where

$$w^g_{s,t} = \frac{1}{|W|^2} \sum_{k\in\{k|s,t\in W_k\}} \left(1 + \frac{\left(I^g_s - \mu_k\right)\left(I^g_t - \mu_k\right)}{\sigma_k^2 + \epsilon}\right). \qquad (4)$$

Note that $|W|$ indicates the window size, and the parameters μ_k and σ_k^2 are the mean and variance of pixel values in window W_k of I^g, respectively. The guided filter is controlled by two parameters: w and ϵ; the former parameter determines the window radius, and the latter controls the extent of smoothing (i.e., a large ϵ produces highly smoothed outputs). As noted [14], guided filtering with $w = \sigma_s$ and $\epsilon = \sigma_r^2$ would produce comparable outputs as the standard bilateral filtering with σ_s and σ_r does, while lower computation costs can be achieved.

2.3. Geodesic Filtering

Without applying explicit spatial kernels, geodesic filtering [12] utilizes only photometric distances during the filtering process. It calculates the output at pixel s by $I'_s = \frac{1}{Z_s}\sum_{t\in\mathcal{N}(s)} g(d_{\text{GF}}(I_s,I_t)\,;\sigma_r)\, I_t$, where Z_s is the normalization factor, and $\mathcal{N}(s)$ contains neighboring pixels of s. For geodesic filtering, the photometric distance between pixels s and t is defined as

$$d_{\text{GF}}(I_s,I_t) = \min_{\phi} \sum_{x,\, x+1\in\phi} \|I_{x+1} - I_x\|, \qquad (5)$$

where ϕ is the path connecting pixels s and t. It is clear that this path is determined by the minimum value of the accumulated differences between adjacent pixel values connecting s and t. This makes geodesic filters adaptive to image context, and thus perform well in image denoising (especially with small σ_r). However, as discussed in Section 3.2, geodesic filters suffer from cross-region mixing in image smoothing tasks (i.e., with larger σ_r). Although Criminisi *et al.* [5] further incorporated predefined spatial distances into (5) for alleviating the above problem, additional efforts would be required for parameter selection in their work.

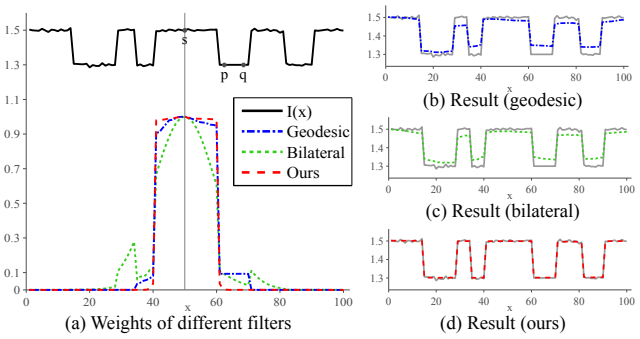


Figure 1: Illustration of cross-region mixing (in 1D) for pixels in textural image regions. The value of pixel x is denoted as $I(x)$, and the dotted lines indicate the weights derived by different filters (we choose $\sigma_s = 10$ for the bilateral filter, and $\sigma_r = 0.1$ for all filters). Note that we intentionally disregard noisy pixels between pixels p and q, so that the characteristics of geodesic filters can be better illustrated.

3. Propagation Filtering

As illustrated in Figure 1, cross-region mixing is a typical problem for existing filters when performing image processing tasks like denoising or smoothing. For example, although bilateral filters measure the photometric distances between pixels for determining the filter weights, their use of explicit spatial filtering kernels would inevitably assign weights to pixels across image regions. On the other hand, geodesic filters utilize image context information for filtering by accumulating the value differences from adjacent pixels. However, when adjacent image regions are of different types of context but noise-free, it would not be able to suppress their effects and thus result in cross-region mixing (e.g., $d_{\text{GF}}(I_s,I_p) \approx d_{\text{GF}}(I_s,I_q)$ in Figure 1).

To suppress undesirable information from adjacent or neighboring pixels during filtering, we propose the propagation filter which aims at taking the context information between image pixels into consideration. Without using any explicit spatial functions, we develop our filter from a probabilistic point of view, as detailed in the following subsections. We will show that our propagation filter essentially cooperates the merits of bilateral and geodesic filtering, while comparable computation costs can be obtained.

3.1. The One-Dimensional Case

Given an input image I, the filtered output I'_s at pixel s produced by our propagation filter is calculated by

$$I'_s = \frac{1}{Z_s} \sum_{t \in \mathcal{N}(s)} w_{s,t}\, I_t, \tag{6}$$

where I_t denotes the value at pixel t in I, and $\mathcal{N}(s)$ indicates the set of neighboring pixels centered at s. We have $w_{s,t}$ as the weight for each pixel t to perform the filtering of I_s, while $Z_s = \sum_{t \in \mathcal{N}(s)} w_{s,t}$ as the normalization factor for ensuring the sum of all $w_{s,t}$ equal to 1.

We first define that pixels y is *photometrically related* to pixel x, or $x \xrightarrow{r} y$, if x and y have similar pixel values. In addition, if y is also adjacent to x, we say that y is *adjacent-photometrically related* to x, or $x \xrightarrow{a} y$. Given these two types of photometric relationships, we define the pixel relationship as follows:

Definition 1. *Suppose there are n singly connected pixels $1, \ldots, n$, in which s and t are two pixels satisfying $1 \le s < t \le n$ without the loss of generality. Pixel t is related to pixel s, or $s \to t$, if and only if $s \to t{-}1$, $t{-}1 \xrightarrow{a} t$, and $s \xrightarrow{r} t$. In addition, each pixel is always self-related, i.e., $s \to s$.*

In other words, for pixel t being related to pixel s, the intermediate pixels between s and t not only need to be photometrically related to s, they are also required to be adjacent-photometrically related to their predecessors (as depicted in Figure 2a). As a result, we derive the filter weight $w_{s,t}$ by the following definition:

Definition 2. *Suppose there are n singly connected pixels, $1, \ldots, n$, and pixels s and t satisfying $1 \le s \le t \le n$. The weight $w_{s,t}$ for filtering pixel s with pixel t is the probability value of t being related to s, i.e., $P(s \to t)$.*

If $t = s$, we have $w_{s,s} = P(s \to s) = 1$. As for $t \neq s$, based on Definitions 1 and 2, we calculate the weight $w_{s,t}$ by the Bayes' rule:

$$\begin{aligned} w_{s,t} &\equiv P(s \to t) = P\Big(s \to t{-}1 \;\wedge\; t{-}1 \xrightarrow{a} t \;\wedge\; s \xrightarrow{r} t\Big) \\ &= P(s \to t{-}1)\; P\Big(t{-}1 \xrightarrow{a} t \;\wedge\; s \xrightarrow{r} t \mid s \to t{-}1\Big) \\ &= w_{s,t-1}\; P\Big(t{-}1 \xrightarrow{a} t \,|\, s \to t{-}1\Big)\; P\Big(s \xrightarrow{r} t \,|\, s \to t{-}1 \wedge t{-}1 \xrightarrow{a} t\Big) \\ &\equiv w_{s,t-1}\; D(t{-}1, t)\;\; R(s,t)\,. \end{aligned} \tag{7}$$

It can be seen that, $P(s \to t{-}1)$ in (7) is replaced by $w_{s,t-1}$, while the third equality holds due to $P(A \wedge B \mid C) = P(A \mid C) P(B \mid A \wedge C)$. Since $t{-}1 \xrightarrow{a} t$ and $s \to t{-}1$ are independent events, we have $P\Big(t{-}1 \xrightarrow{a} t \,|\, s \to t{-}1\Big) = P\Big(t{-}1 \xrightarrow{a} t\Big) \equiv D(t{-}1, t)$, denoting the *adjacent photometric relationship* between pixels t and $t{-}1$. By assuming that

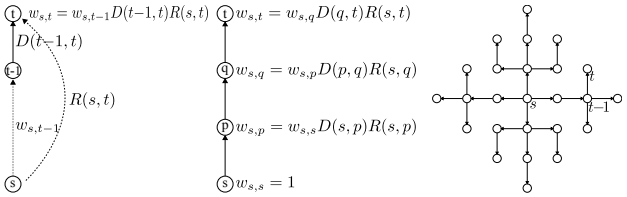


Figure 2: Illustration of propagation filtering. (a) the definition of filtering weight $w_{s,t}$, (b) the calculation of $w_{s,t}$, and (c) the pattern for performing 2D filtering with $d = 3$ pixels.

the probability value of two adjacent pixels being photometric related is proportional to the value of a Gaussian function of their pixel value difference, we define

$$D(x, y) = g(\|I_x - I_y\|; \sigma_a) = \exp\left(\frac{-\|I_x - I_y\|^2}{2\sigma_a^2}\right), \tag{8}$$

where I_x is the value of pixel x, and $\|I_x - I_y\|$ measures the Euclidean distance between the corresponding pixel value difference.

As for $R(s,t)$ in (7), we have $R(s,t) = P\Big(s \xrightarrow{r} t \,|\, s \to t{-}1 \wedge t{-}1 \xrightarrow{a} t\Big)$, which calculates the *photometric relationship* between pixels s and t. With $s \to t{-}1$ and $t{-}1 \xrightarrow{a} t$, pixel t is viewed as adjacent to pixel s, and thus $R(s,t)$ is measured as the adjacent-photometric relationship between s and t. As a result, we define

$$R(x, y) = g(\|I_x - I_y\|; \sigma_r) = \exp\left(\frac{-\|I_x - I_y\|^2}{2\sigma_r^2}\right). \tag{9}$$

For simplicity, we choose $\sigma_a = \sigma_r$, and thus $D(\cdot) = R(\cdot)$. We use Figure 2a and 2b to illustrate how to calculate $w_{s,t}$.

Based on the above definitions, we see that a large weight $w_{s,t}$ not only needs pixels s and t to have a strong photometric relationship (i.e., similar pixel values), it also needs large $w_{s,t-1}$ and $D(t{-}1, t)$ values for the intermediate pixels between s and t. That means, if any pixel along the path connecting pixels s and t is unrelated to either of them, t will be viewed as unrelated to s. In other words, a small $w_{s,t}$ will be resulted. This is the reason why our propagation filter is able to reflect image context information when performing filtering.

3.2. Comparison to Bilateral and Geodesic Filtering

We now compare propagation filtering with bilateral and geodesic filtering. For our propagation filters, we derive the filter weights from a probability point of view. To be more precise, by expanding (7), we obtain the filtering weights $w_{s,t}$ in (6) regarding the pixel distances as:

$$w_{s,t} = g\big(\, d^a_{\text{PF}}(I_s, I_t);\, \sigma a\big)\;\; g\big(\, d^r_{\text{PF}}(I_s, I_t);\, \sigma r\big) \tag{10}$$

$$d^a_{\text{PF}}(I_s, I_t) = \sqrt{\sum_{x, x+1 \in \phi} \|I_{x+1} - I_x\|^2} \tag{11}$$

$$d^r_{\text{PF}}(I_s, I_t) = \sqrt{\sum_{x \in \phi} \|I_x - I_s\|^2}, \tag{12}$$

Note that ϕ is the path connecting pixels s and t. Our propagation filter inherits merits from both bilateral and geodesic filters while compensating their deficiencies. That is, instead of using explicit spatial kernels, we have $d^a_{\text{PF}}(I_s, I_t)$ utilize intermediate adjacent pixel information between s and t, which is similar to the photometric distances applied in geodesic filtering (5) and recursive bilateral filtering (3).

The deficiency of geodesic filters is complemented by $d^r_{\text{PF}}(I_s, I_t)$, which measures the photometric distance between pixels along the path of interest. As noted earlier, geodesic filters cannot suppress effects from nearby image regions well, especially when only negligible noise is observed in those regions (e.g., $d_{\text{GF}}(I_s, I_p) \approx d_{\text{GF}}(I_s, I_q)$ in Figure 1). Instead, our propagation filter is able to observe large $d^r_{\text{PF}}(I_s, I_p)$ and $d^r_{\text{PF}}(I_s, I_q)$ and have $w_{s,q} < w_{s,p}$. This would alleviate cross-region mixing problems, and this is the reason why propagation filters better discriminate between image regions with different context information.

To sum up, incorporating both $d^a_{\text{PF}}(I_s, I_t)$ and $d^r_{\text{PF}}(I_s, I_t)$ makes our propagation filters more adaptive to image context information. As verified later, this allows one to obtain satisfactory filtering results than bilateral and geodesic filters do in several image processing tasks.

3.3. The Two-Dimensional Case

For image filtering, we now extend the above process to two-dimensional scenarios. Given pixels s and t in an image, we need to first determine a path connecting s and t for deriving the weight $w_{s,t}$ accordingly. Although one can apply the Dijkstra's shortest path algorithm [6] to determine the path with the largest weight $w_{s,t}$ (like geodesic filters [12] do), the resulting computation complexity will be $O(W \log(W))$. In other words, filtering an image with a total of N pixels will cost $O(NW \log(W))$, which is computationally expensive when W or N is large.

Instead, similar to [24] and [3], we consider a particular 2D pattern for filtering all pixels in an image. The pattern we propose is shown in Figure 2c. It can be seen that, the path would be a straight line connecting pixels s and t, if these two pixels are horizontally or vertically aligned. If pixels s and t are not simply horizontally or vertically connected, we determine the path based on their *Manhattan* distance. That is, if the Manhattan distance between s and t is an odd number, we choose the path for traversing from that pixel to its predecessor (e.g., from t to $t-1$) in the vertical direction; otherwise, the horizontal path will be chosen.

By using this 2D pattern for filtering, the computation complexity of our propagation filter will be reduced to $O(NW)$. It is the same as that of a standard bilateral filter, and smaller than that of a geodesic filter (requiring $O(NW \log(W))$ operations). For example, our filter with $w = 10$ used 2.4 seconds to filter one million pixels on a Intel Core i7 machine. In contrast, a geodesic filter [12] with the same setting took 23 seconds. It is worth noting that, one cannot directly apply methods like frequency transform [18]) or recursive implementations [24] for propagation filtering, since the derivation of the filter weights does not affect those for other pixels in the same image. However, due to this exact property, our propagation filtering can be easily parallelized and speedup by using multi-core processing techniques.

3.4. Propagation Filtering as a Robust Estimator

We now show that our proposed propagation filter can be viewed as a one-step estimator, which aims at minimizing the error between the filtered and desirable outputs (e.g., denoised or smoothed images). To start, we first transform our filtering algorithm of (6) into the following formulation:

$$I'_s = \frac{1}{Z_s} \sum_{t \in \mathcal{N}(s)} w_{s,t}\, I_t = I_s - \frac{1}{Z_s} \sum_{t \in \mathcal{N}(s)} w_{s,t}\, (I_s - I_t). \tag{13}$$

The above equation is effectively a *gradient descent* solver optimizing the objective function f, whose gradient at pixel s is derived as:

$$\nabla f = \sum_{t \in \mathcal{N}(s)} w_{s,t}\, (I_s - I_t)\,. \tag{14}$$

With this gradient solver, the optimization problem can be recovered as:

$$\min_I \sum_{s \in \Omega} \sum_{t \in \mathcal{N}(s)} \int w_{s,t}\, (I_s - I_t)\; \mathrm{d}(I_s - I_t), \tag{15}$$

where Ω denotes the pixel set of the input image. According to Definition 2, the weight $w_{s,t}$ represents the probability of pixel t related to s. As noted earlier, our propagation filter assumes that related pixels exhibit similar pixel values. Therefore, we apply $w_{s,t}$ as the belief $P(I_s - I_t)$ and have $\int w_{s,t}\, (I_s - I_t)\; \mathrm{d}(I_s - I_t) = \int P(s \to t)\, (I_s - I_t)\; \mathrm{d}(I_s - I_t) \equiv \int P(I_s - I_t)\, (I_s - I_t)\; \mathrm{d}(I_s - I_t)$, which indicates the *expectation* of the difference between pixels s and t with probability $P(s \to t)$. Now, we can rewrite (15) as:

$$\min_I \sum_{s \in \Omega} \sum_{t \in \mathcal{N}(s)} \mathrm{E}\,[I_s - I_t]\,. \tag{16}$$

From (16), we see that our propagation filtering is a one-step estimator, which minimizes the expected difference between each filter pixel and its related ones (i.e., the neighboring ones with similar context information).

As noted in [7, 13], bilateral filtering can also be considered a one-step estimator for achieving the above goal. However, as discussed in the beginning of Section 3, bilateral filtering is more likely to encounter the problem of cross-region mixing due to its use of predetermined spatial kernel functions, and thus limits its ability in adapting the image context. Later in Section 4 (and in the supplementary materials), we will further verify the effectiveness and superiority of our propagation filter.

3.5. Propagation Filtering as Belief Propagation

We now show that propagation filtering can be viewed as belief propagation with a Bayesian network. As depicted in Figure 2c, our 2D path pattern for image filtering is a polytree (i.e., a directed acyclic graph). We now show that our filtering process is equivalent to the polytree algorithm [19].

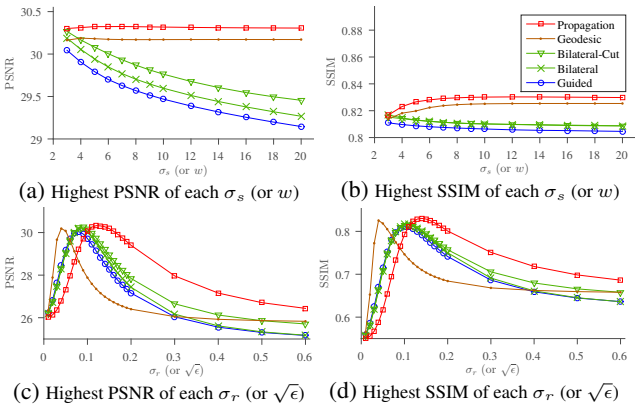
Let X_t as the random variable representing that pixel t is related to pixel s. A Bayesian network can be constructed using our proposed 2D pattern, which reflects the dependency between its nodes. By Definition 1, we have s and $t{-}1$ as the parent nodes of t, and $t{+}1$ as its child node. Since there only exists a single directed path from t to $t{+}1$, the probability $P(X_t)$ can be expressed as $P(X_t) = P(Xs)P(X_{t-1} \mid Xs)P(X_t \mid Xs \wedge X_{t-1})$, where $P(Xs)$ and $P(X_{t-1} \mid Xs)$ indicate the beliefs of Xs and X_{t-1}.

We now show that $P(X_t)$ is effectively the weight $w_{s,t}$ as determined in our propagation filter. According to Definition 2, since pixel s is to be filtered, we have $P(Xs) = w_{s,s} = 1$ and $P(X_{t-1} \mid Xs) = P(X_{t-1}) = w_{s,t-1}$. Based on the same definition, we calculate $P(X_t \mid Xs, X_{t-1})$ by $P(t{-}1 \xrightarrow{a} t \wedge s \xrightarrow{r} t \mid s \to t{-}1)$, and thus $P(X_t) = w_{s,t-1}P(t{-}1 \xrightarrow{a} t \wedge s \xrightarrow{r} t \mid s \to t{-}1)$, which is the same as (7). From the derivations, we have $P(X_t) = w_{s,t}$, which verifies that our propagation filtering can be regarded as the belief propagation polytree algorithm.

3.6. Speedup

In addition to parallelization processing by multi-core techniques, further acceleration for propagation filtering can be achieved by early cutoff in the derivation of the filter weights. Recall that the weights of propagation filters are calculated by (7). If the weight of pixel t for filtering s is small, its successors (e.g., $t{+}1$) will also exhibit low weight values based on the derivations provided in Section 3.1. Therefore, one can apply a threshold into the filtering process for allowing early termination of weight calculation. Moreover, it would further prevent undesirable filtering effects such as cross-region mixing.

To support the above speedup strategy, we observe image statistics from a thousand of images randomly collected from the Internet[1]. Using $d = 20$ for performing propagation filtering, about 30% of the filter weights for the image pixels are lower than 0.001 if $\sigma_r = 0.1$ is chosen. If a smaller $\sigma r = 0.01$ is selected (which better preserves image details during image smoothing), approximately 80% of the filter weights are lower than 0.001. In other words, if 0.001 is used as the threshold for early cutoff, 80% of the original computation time can be saved.



(a) Highest PSNR of each σ_s (or w) (b) Highest SSIM of each σ_s (or w)

(c) Highest PSNR of each σ_r (or $\sqrt{\epsilon}$) (d) Highest SSIM of each σ_r (or $\sqrt{\epsilon}$)

Figure 3: PSNR and SSIM comparisons for image denoising. (a) and (b) show the best PSNR and SSIM results for each σ_s (or w) value with the optimal σ_r (or $\sqrt{\epsilon}$), while (c) and (d) present those for each σ_r (or $\sqrt{\epsilon}$) value using the optimal σ_s (or w).

4. Applications and Experimental Results

Denoising

We now apply our propagation filter to a variety of applications in computer vision and graphics. We first consider the task of image denoising, and we choose to add white Gaussian noise with the standard deviation of 0.05 to the input image[2]. We compare the denoised outputs produced by bilateral, guided, geodesic, and our propagation filters. Note that the pixel values of gray-scale images in our experiments are in the range of [0, 1].

For parameter selections, we assign the same σ_r for bilateral, geodesic, and propagation filters. This would allow these filters to exhibit the same ability in determining the photometric relationships between image pixels. We vary σ_s of the spatial Gaussian function for the bilateral filter, and discuss the corresponding results. On the other hand, the Manhattan distance $w = \sigma_s$ is applied as the window radii for both propagation and geodesic filters. In addition, we consider *bilateral-cut* with the same σ_s and σ_r choices, and perform filtering with the above window setting. For the guided filter, we choose $w = \sigma_s$ and $\epsilon = \sigma_r^2$ for producing comparable results as the bilateral filter does (as suggested in [14]). Note that no early cutoff for accelerating the propagation filtering is performed in the experiments.

[1] Flickr.com

[2] By courtesy of http://www.flickr.com/photos/jon-luke/4711647414.

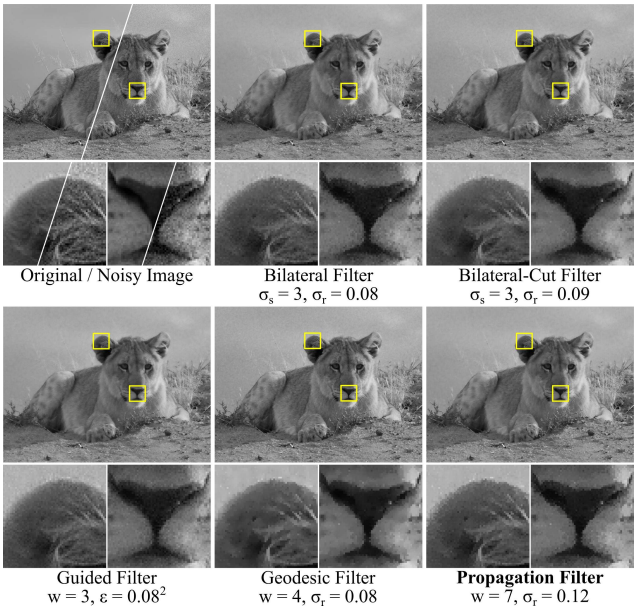


Figure 4: Example results of image denoising (with the highest PSNR for each filter).

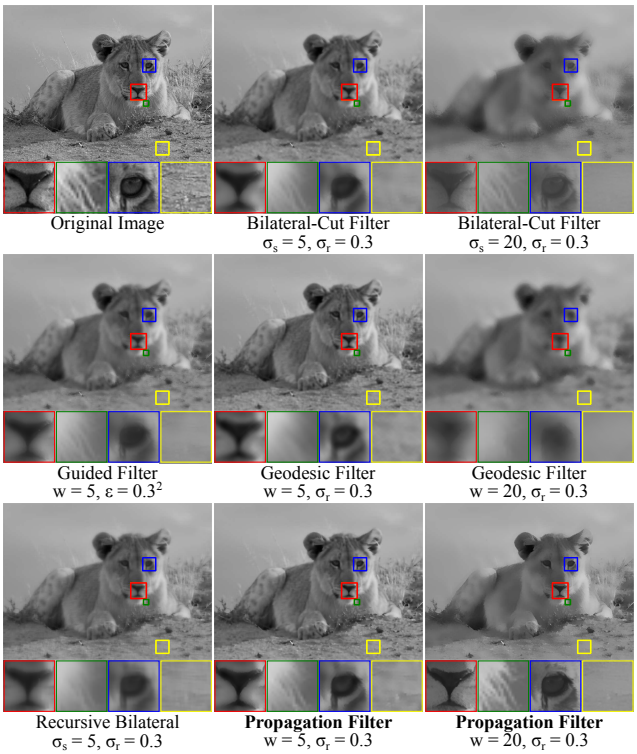


Figure 5: Example results of image smoothing.

We apply the metrics of PSNR and SSIM for quantitative evaluation. For fair comparisons, we always fix one parameter (e.g., σ_S or w) for all filters and vary the other one (e.g., σ_r or ϵ) for reporting the best denoising performance. The highest PSNR and SSIM values are presented in Figure 3. From Figures 3a and 3b, we see that our propagation filter achieved satisfactory PSNR or SSIM values across different w, and it consistently outperformed bilateral and guided filters. It can be seen that, while larger w (or σ_S) would cause cross-region mixing for existing filters, our use of pixel photometric relationships for filtering is able to alleviate such undesirable artifacts.

From Figures 3c and 3d, we see that bilateral and guided filters produced higher PSNR and SSIM values than ours did when σ_r was below 0.1. This is because that, when a very small σ_r^2 is set (e.g., lower than the variance of the added noise), our propagation filter will consider the corrupted pixels as outliers, due to the observed low photometric relationships. This effectively terminates the propagation process and reduces the neighborhood size. This characteristic limits the denoising performance of propagation filters with small σ_r. Nevertheless, when reasonable σ_r values were chosen, our propagation filter achieves high PSNR and SSIM values. It can also be seen that even though geodesic and propagation filters have similar denoising performance (i.e. the highest PSNR value), when σ_r is large, the geodesic filter produces low PSNR and SSIM values. The results verify our discussions earlier in the section that the geodesic filter cannot prevent cross-region mixing especially when σ_r is large. This will also be shown in our smoothing results. Figure 4 compares the denoised outputs of different filters with the highest PSNR. It can be seen that, more image details were successfully preserved by our propagation filter. More examples can be found in the supplemental materials.

To further compare our approach with optimization-based image filters, we consider recent approaches of [10, 23]. With the same test inputs and experiment settings, degraded performances were observed (-0.5 and -2.3 in PSNR, and -0.01 and -0.03 in SSIM for [23] and [10], respectively). We note that, the above optimization-based image filters focus on decomposing the input image into base and detailed layers. Such strategies might not be preferable for denoising or smoothing, since solving these tasks require the filtering algorithms to preserve image local details. In [11], 2D image filtering is approximated by iterating the 1D ones via isometric transforms. Our filter also achieved improved results when comparing to [11] (0.76 in PSNR and 0.026 in SSIM). We believe that this is due to the 2D filtering approximation considered in [11], which tends to encounter cross-region mixing problems during the denoising process.

Image Smoothing

Extended from image denoising, image smoothing further suppresses image detailed information (e.g., highly textural regions), while strong image context can be preserved (e.g., edges). In addition to the grayscale image of Fig-

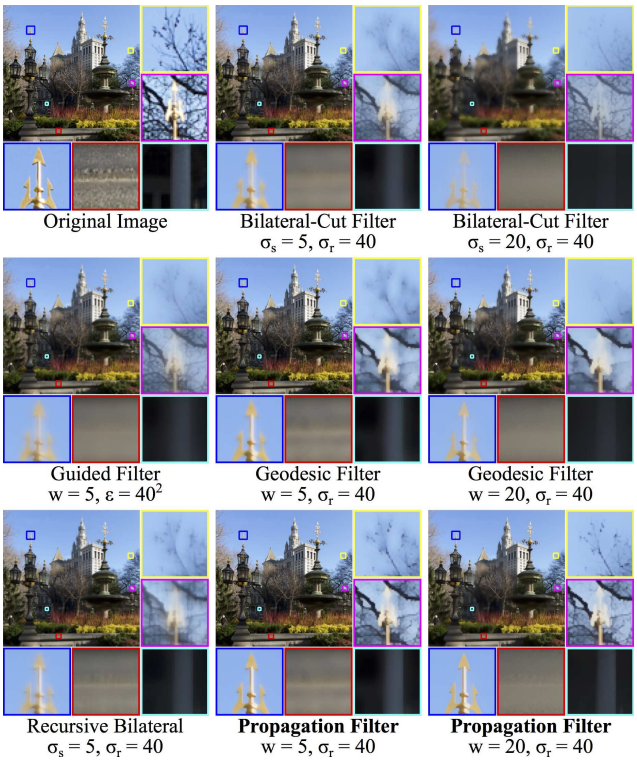


Figure 6: Examples of color image smoothing.

ure 4, smoothing of a color image[3] is also considered. We note that, when performing color image filtering, the filter weights are calculated in the CIELAB space. The values of intensity and color channels in this space are in the ranges of [0, 100] and [-128,127], respectively.

Figures 5 and 6 show the smoothing results of different images. As noted earlier, we chose the same σ_r (or $\sqrt{\epsilon}$) for all filters for fair comparisons. Compared with our propagation filters, we see that all other filters were not able to properly preserve image edges. As depicted in Figure 6, while larger σ_S obviously resulted in eroded building spires for bilateral and geodesic filters due to cross-region mixing (see image regions in blue and magenta boxes in Figure 6 for examples), the use of smaller σ_S was still not able to preserve edges in such highly textural regions. Examples in supplemental materials show that guided and recursive bilateral filters have similar problems. It is worth repeating that since our propagation filter utilizes both intermediate adjacent pixel information and direct photometric relationships with the center pixel, it adapts well to image context. This explains why better image smoothing outputs can be expected even the image contains both smooth and highly textural regions.

[3]By courtesy of http://www.flickr.com/photos/hhoyer/7105107291

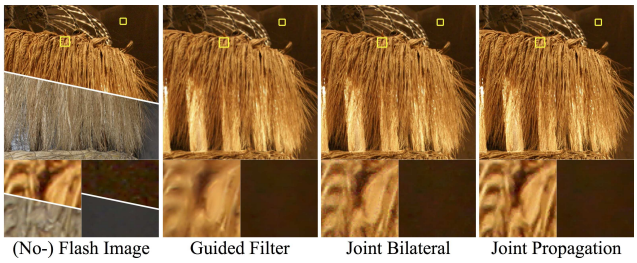


Figure 7: Flash/no-flash denoising. Input images are from [20]. Parameter choices for the bilateral filter: $\sigma_s = 3$ and $\sigma_r = 10^{-3}$ (as [20]), the guided filter: $w = 3$ and $\epsilon = (10^{-5})^2$, and our propagation filter: $w = 12$ and $\sigma_r = 0.01$.

Flash/No-Flash Denoising

Generally, flash images contain more image details, but their colors are less vivid due to extreme illumination conditions. Petschnigg *et al.* [20] proposed the technique of joint bilateral filtering, which denoises a no-flash image under the guidance of its flash version. In other words, with the goal of preserving image characteristics, it calculates filter weights from the flash image, and filtering is performed over the no-flash version.

The same operation can be applied to propagation filtering. Figure 7 presents and compares the results produced by bilateral, guided, and propagation filters. It can be seen that, although bilateral and guided filters were able to disregard image noise in dark regions, they failed to preserve image details as our propagation filters did. More example results can be found in our supplementary materials.

From the above examples, we note that our filtering algorithm can be applied and integrated into other filters, which utilize additional reference images for solving particular tasks. For example, the bilateral filtering process in [4] can be replaced by our propagation filtering algorithm. Since ours better alleviates cross-region mixing problems than the standard bilateral filter does, improved results can be expected if such replacement is deployed.

Image Fusion

Image fusion aims at integrating multiple images to one single output, so that the resulting image would contain more comprehensive information than each of the inputs does. A popular example of image fusion is the combination of multi-focus images. Li *et al.* [15] proposed an algorithm which is able to fuse images, with the weights determined by the corresponding saliency or detailed image information. Based on the algorithm of [15], we consider two images with different focuses [4] and compare the fused outputs produced by bilateral, guided, and our propagation filters. Figure 8 shows the input images and the filtered outputs.

[4]By courtesy of http://www.imgfsr.com/ifsr˙ifs1.html

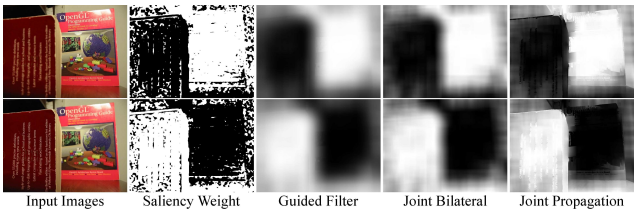


Figure 8: Input images and saliency weights recovered by the corresponding images. Parameter selection for the guided filter: $w = 45$ and $\epsilon = 0.3$, the bilateral filter: $\sigma_s = 45$ and $\sigma_r = \sqrt{0.3}$, and our propagation filter: $w = 45$ and $\sigma_r = \sqrt{0.3}$.

We note that, the experiment settings, including parameter selection for the filters considered, were the same as those in [15]. From this figure, we can see that the weights recovered by our propagation filter were more consistent with image context (e.g., salient object regions and boundaries) without smoothing such regions. Therefore, we verify that our propagation filter can be applied to solving the task of image fusion.

High Dynamic Range Compression

Finally, we apply our propagation filter for high-dynamic-range (HDR) compression. Using the algorithm of Durand and Dorsey [7], an input image can be divided into base and detail layers. The former is acquired by performing the bilateral filtering on the input image, while the latter is produced by subtracting the base layer from the input image. By compressing only the base output and adding the pixel information from the detail layer, one can reduce the dynamic range of pixel values, and the image details can be better preserved.

We now replace the bilateral filter by our propagation filter in the algorithm of [7], and we compare the HDR compression results in Figure 9. Comparing Figures 9a and 9b, the compressed output using bilateral filtering contained over-bright and dark regions (e.g., the logo (in yellow box), face (in blue box) and poster (in red box) in Figure 9a). This is because that bilateral filters are more likely to encounter cross-region mixing and produce over-smooth base images. As a result, information extracted in the detail layer will be over-emphasized and thus result in saturated image outputs.

As shown in Figure 9b, our propagation filter produced satisfactory compression performance while preserving image details. It is worth noting that, instead of using a fixed σr (e.g., Figure 9b), we can determine $\sigma r(s)$ for filtering each pixel s. This allows us to exploit image context information during the filtering process. An example output is shown in Figure 9c, in which we calculated $\sigma r(s)$ based on the standard deviation of neighboring pixel values of each pixel s to be filtered, and improved filtering results can be observed.

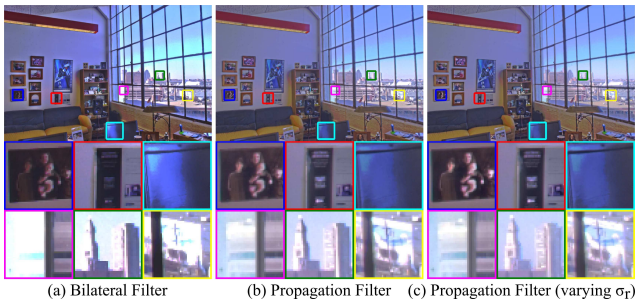


Figure 9: Examples of HDR compression. (a) Result of [7] (via bilateral filtering), (b) output using our propagation filter with $w = 16$ and $\sigma_r = 2.5$, and (c) output with propagation filtering with $w = 16$ and varying σ_r (i.e., $\sigma_r(s) = 2.5$ std$(\mathcal{N}(s))$).

5. Conclusion

In this paper, we presented the propagation filter as a local filtering operator, which aims at smoothing over images while preserving image context information. While sharing the same goal of preserving image details as those of bilateral and guided filters, our propagation filter is based on the photometric relationship observed between image pixels. Without using explicit spatial filtering functions, we are able to alleviate cross-region mixing problems during filtering, and thus image characteristics can be better preserved. In addition to comparable computation complexity as the standard bilateral filter, we showed that the process of propagation filtering is equivalent to robust estimation and belief propagation, which provide theoretical supports to our proposed filter. Finally, a variety of applications computer vision and graphics verified the effectiveness of our propagation filter, which was shown to outperform existing image filters in terms of both quantitative and qualitative evaluations.

Acknowledgement This work is supported in part by the Ministry of Science and Technology of Taiwan via MOST103-2221-E-001-021-MY2 and NSC102-2221-E-001-005-MY2.

References

[1] M. Aleksic, M. Smirnov, and S. Goma. Novel bilateral filter approach: Image noise reduction with sharpening. In *Electronic Imaging*, 2006. 1

[2] S. Bae, S. Paris, and F. Durand. Two-scale tone management for photographic look. *ACM Transactions on Graphics (TOG)*, 2006. 1

[3] X. Chen, S. B. Kang, J. Yang, and J. Yu. Fast patch-based denoising using approximated patch geodesic paths. In *CVPR*, 2013. 4

[4] H. Cho, H. Lee, H. Kang, and S. Lee. Bilateral texture filtering. *ACM Transactions on Graphics (TOG)*, 2014. 7

[5] A. Criminisi, T. Sharp, C. Rother, and P. Perez. Geodesic image and video editing. *ACM Transactions on Graphics (TOG)*, 2010. 1, 2

[6] E. W. Dijkstra. A note on two problems in connexion with graphs. *Numerische mathematik*, 1959. 4

[7] F. Durand and J. Dorsey. Fast bilateral filtering for the display of high-dynamic-range images. *ACM Transactions on Graphics (TOG)*, 2002. 1, 5, 8

[8] E. Eisemann and F. Durand. Flash photography enhancement via intrinsic relighting. *ACM Transactions on Graphics (TOG)*, 2004. 1

[9] M. Elad. Retinex by two bilateral filters. In *Scale Space and PDE Methods in Computer Vision*. 2005. 1

[10] Z. Farbman, R. Fattal, D. Lischinski, and R. Szeliski. Edge-preserving decompositions for multi-scale tone and detail manipulation. *ACM Transactions on Graphics (TOG)*, 2008. 6

[11] E. S. Gastal and M. M. Oliveira. Domain transform for edge-aware image and video processing. *ACM Transactions on Graphics (TOG)*, 2011. 6

[12] J. Grazzini and P. Soille. Edge-preserving smoothing using a similarity measure in adaptive geodesic neighbourhoods. *Pattern Recognition*, 2009. 1, 2, 4

[13] F. R. Hampel, E. M. Ronchetti, P. J. Rousseeuw, and W. A. Stahel. *Robust statistics: the approach based on influence functions*. Wiley.com, 2011. 5

[14] K. He, J. Sun, and X. Tang. Guided image filtering. In *ECCV*. 2010. 1, 2, 5

[15] S. Li, X. Kang, and J. Hu. Image fusion with guided filtering. *IEEE Transactions on Image Processing*, 2013. 1, 7, 8

[16] C. Liu, W. T. Freeman, R. Szeliski, and S. B. Kang. Noise estimation from a single image. In *CVPR*, 2006. 1

[17] J. Lu, K. Shi, D. Min, L. Lin, and M. N. Do. Cross-based local multipoint filtering. In *CVPR*, 2012. 1

[18] S. Paris and F. Durand. A fast approximation of the bilateral filter using a signal processing approach. In *ECCV*. 2006. 4

[19] J. Pearl. *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*. Morgan Kaufmann Publishers Inc., 1988. 5

[20] G. Petschnigg, R. Szeliski, M. Agrawala, M. Cohen, H. Hoppe, and K. Toyama. Digital photography with flash and no-flash image pairs. In *ACM Transactions on Graphics (TOG)*, 2004. 1, 7

[21] K. Subr, C. Soler, and F. Durand. Edge-preserving multiscale image decomposition based on local extrema. *ACM Transactions on Graphics (TOG)*, 2009. 1

[22] C. Tomasi and R. Manduchi. Bilateral filtering for gray and color images. In *ICCV*, 1998. 1

[23] L. Xu, C. Lu, Y. Xu, and J. Jia. Image smoothing via L_0 gradient minimization. *ACM Transactions on Graphics (TOG)*, 2011. 6

[24] Q. Yang. Recursive bilateral filtering. In *ECCV*. 2012. 1, 2, 4