CGI2015 manuscript No.
(will be inserted by the editor)

Two-Level Joint Local Laplacian Texture Filtering

Abstract Extracting the structure component from an image with textures is a challenging problem in image processing. This paper presents a novel structure-preserving texture filtering approach based on the two-level local Laplacian filter. The new method is developed by introducing local Laplacian filters into the joint filtering. Our study shows that local Laplacian filters can also be used for texture smoothing by defining a special remapping function, which is closely related to the joint bilateral filtering. This finding leads to a variant of the joint bilateral filter, we call it *unnormalized joint bilateral filter*, which produces smooth edges while preserving color variations. Our filter shares similar advantages with the joint bilateral filter, such as simple to implement and easy to understand. Experiments demonstrate that the new filter can produce satisfactory filtering results with the properties of texture smoothing, smooth edges, and edge shape-preserving. We compare our method with the state-of-the-art methods to demonstrate its improvements, and apply this filter to a variety of image editing applications.

Keywords Texture filtering · structure extraction · local Laplacian filters · guidance image · unnormalized joint bilateral filter

1 Introduction

Edge-preserving image smoothing is very important and useful in many image applications, e.g., detail manipulation, abstraction, tone mapping, image composition, etc. It seeks to decompose an image into structure and detail components. Most existing works, such as the bilateral filter [22], weighted least squares filtering [7], L_0 smoothing [25] and local Laplacian filters [16], focused on separating structures from details with the capability of edge-preserving. These algorithms usually depend on gradient magnitude, pixel intensity and Laplacian pyramid coefficients to obtain satisfactory edge-preserving filtering results. While humans easily recognize the structures of an image which contains rich textures, it is difficult for a computer to automatically extract its structures by texture smoothing. Most edge-preserving filtering methods do not explicitly address the structure-preserving texture smoothing problem. Textures may be considered as strong edges if we apply these approaches straightforwardly to texture smoothing. As a result, textures are preserved instead of being smoothed and unsatisfactory results may arise.

To solve this challenging problem, previous works [3, 5, 11, 20, 21, 26, 28] have made much effort to generate structure-preserving texture smoothing results. They suppress textures to obtain the global structures of an image using different strategies, such as the weighted least squares optimization or the joint bilateral filtering schemes. These solutions can achieve good separation results. However, rich color variations of the input may be flattened somewhat or structure edges of the input may be damaged by these filters. For example, after the bilateral texture filtering [5], smooth edges in the original image may show jagged transitions in the resulting structure component, as shown in Fig. 1(e). When applying such structure images to image applications (e.g. detail enhancement), undesirable results may arise at structure edges. Thus, in order to achieve satisfactory results, smooth edges in the extracted structure component should be preserved, and the edge shapes should be as similar to those of the input as possible.

In this paper, we present a new structure-preserving texture filtering method which can preserve edge shapes

(a) (b) (c) (d) (e) (f)

Fig. 1 Two-level joint local Laplacian texture filtering. From left to right: (a) Input image; (b) Xu et al. [26]; (c) Karacan et al. [11]; (d) Zhang et al. [28]; (e) Cho et al. [5]; (f) our method. In terms of color variations, our method performs better than Xu et al. [26]. Our method can preserve partial structures of the fish teeth, which are removed in both results of Zhang et al. [28] and Cho et al. [5]. Zhang et al. [28] produces halo artifacts near strong structure edges which are reduced in our result. Compared to Cho et al. [5], our method produces smoother structure edges.

and color variations of the input image simultaneously. Our algorithm builds upon local Laplacian filters. While local Laplacian filters can yield high-quality edge-aware filtering results using a Laplacian pyramid, it is still unclear whether they are appropriate for strong texture smoothing. Inspired by Aubry et al. [1], we redefine the remapping function of local Laplacian filters by introducing a guiding image. With this in hand, we find that the two-level joint local Laplacian filter is related to the joint bilateral filter and shares similar advantages of the joint bilateral filter. Moreover, we design an unnormalized joint bilateral filter, which essentially is a spatial interpolation-based method between the input image and the result of the joint bilateral filter. The spatial interpolation blending of this filter guarantees that the filtering results own the aforementioned properties, as shown in Fig. 1. Previous works [5, 20, 28] have verified that the joint bilateral filtering schemes can effectively suppress textures. Therefore, our approach can also be employed for structure-preserving texture filtering.

The contributions of our paper include:

- A joint filtering method based on the two-level local Laplacian filter: We introduce a guidance image into the definition of the remapping function of the two-level local Laplacian filter. This allows us to use this filter to achieve joint filtering.
- An unnormalized joint bilateral texture filter: We analyze the similarities between the two-level joint local Laplacian filter and the original joint bilateral filter. Based on this analysis, we develop a new texture smoothing filter.
- An interpolation-based scheme of texture filtering: We further find that the unnormalized joint bilateral texture filter is equivalent to a spatially varying blending between the input image and the output of the joint bilateral filter. The interpolation behavior ensures that our filter can produce smooth edges and preserve the original edge shapes.

2 Related Work

Edge-aware filtering The average-based method is a main kind of edge-aware filters. From a computational point of view, these methods use weighted average schemes to eliminate image details, including anisotropic diffusion [17], bilateral filter [22], joint bilateral filter [13, 18], guided filter [10], and geodesic filter [6]. The bilateral filter [22] is one of the most widely employed edge-aware filters since it preserves important image structures while removing fine-scale details. The output of the bilateral filter is a nonlinear weighted average of the input in the spatial neighborhood considering the spatial distance and the difference of the range value simultaneously. The joint/cross filter [13, 18] smoothes an image by using a guidance image to compute the difference of the range value. However, the bilateral filter tends to blur over more edges as the scale of the extracted details increases.

To overcome the limitation, Farbman et al. [7] introduced weighted least squares to achieve edge-preserving multi-scale image decomposition. Further, a few edge-aware filters, including edge-avoiding wavelets [8], local histogram filters [12], local Laplacian filters [16], domain transform [9], L_0 gradient minimization [25], and fast local Laplacian filters [1], were proposed to smooth fine-scale details while preserving image structures. Paris et al. [16] proposed a set of image filters

called local Laplacian filters to achieve artifact-free edge-preserving results based on the Laplacian pyramid. Aubry et al. [1] further considered a wider class of remapping functions for local Laplacian filters, and speeded up local Laplacian filters. Our work is closely related to these two works. The main difference between our work and theirs is that we aim at structure-preserving texture smoothing which is not addressed in these two works.

In addition to the above-mentioned methods, median filters that compute median [24] or weighted median [15,29] in local patches and the local mode filter [23], can also remove high-contrast details of an image.

Structure-preserving texture smoothing Rudin et al. [19] adopted total variation (TV) to successfully eliminate arbitrary textures of irregular shapes. Further, some improved TV models [2,4,26,27] with different norms were proposed to make image decompositions better. Buades et al. [4] computed a local total variation and used a simple nonlinear filter pair to decompose an image into structure and texture components. Xu et al. [26] used relative total variation measures to achieve better separations between structure edges and textures. Subr et al. [21] defined details as oscillations between local extrema and smoothed out fine-scale oscillations. Karacan et al. [11] presented a patch-based algorithm that uses region covariances for structure-preserving image smoothing. However, it is time-consuming and may result in overblurred edges. Su et al. [20] applied an iterative asymmetric sampling degenerative scheme to suppress textures, then used an edge correction operator and a joint bilateral filter to obtain edge-preserving texture suppression results.

Recently, Cho et al. [5] presented a bilateral texture filter based on the idea of patch shift that captures the texture information from the most representative texture patch clear of prominent structure edges. They constructed the guidance image via patch shift on each pixel and employed the joint bilateral filter to obtain the smoothed output. Zhang et al. [28] used a simple iterative joint bilateral filter model to smooth the image. Bao et al. [3] proposed a weighted-average filter called tree filter to achieve strong image smoothing with a minimum spanning tree. Since our method is a variant of the joint bilateral filter, our algorithm can also be classified as the joint bilateral filter.

3 Our Algorithm

We start from simply summarizing the basic principle of local Laplacian filters and then introduce our algorithm in detail. Given a 2D input image I, local Laplacian filters first use a pixel-wise filter with the remapping function $rp(i)$ to generate a transformed image, then compute the Laplacian pyramid L of this image and use the coefficient in the pyramid L as the value of the output pyramid coefficient. Finally, the output pyramid is collapsed to obtain the result.

3.1 Two-level joint local Laplacian filter

Aubry et al. [1] defined the space of remapping functions for local Laplacian filters as the form:

$$r(i) = i - (i-g)f(i-g), \tag{1}$$

where f is a continuous function, i represents the pixel of the input I and g denotes the coefficient of the Gaussian pyramid. Remapping functions are the same as those of Paris et al. [16] when $f(i-g) = (i-rp(i))/(i-g)$ where rp denotes the remapping functions defined by Paris et al. [16].

We modify the space of remapping functions r by introducing a guidance image M:

$$r(i) = i - (i-g)f(i_m - g_m), \tag{2}$$

where $g_m = G_l[M](x,y)$ denotes the coefficient of the Gaussian pyramid at level l and position (x,y) of the guidance image M. We will discuss the definition of the guidance image M in Section 3.3.

For the two-level filter, we need to compute the two levels $L_0[J]$ and $L_1[J]$ of the Laplacian pyramid of the output image J. We assume the residual $L_1[J]$ of the output remains unprocessed resembling Aubry et al. [1], that is, $L_1[J] = L_1[I]$. The 0th level of the Laplacian pyramid of the output image J is computed as the difference between the transformed image $r(I)$ and the corresponding low pass filtered image. That is,

$$L_0[J](p) = r(I_p) - [\overline{G}_{\sigma_p} * r(I)](p), \tag{3}$$

where p denotes a pixel at the position (x,y), $\overline{G}_{\sigma_p}$ is the normalized Gaussian kernel of variance σ_p^2 used to build the pyramids, and $*$ indicates the convolution operator. For the finest level of the pyramid, we have $L_0[I] = I - \overline{G}_{\sigma_p} * I$, $g = I_p$ and $g_m = M_p$. Substituting our new definition of the remapping function into the above expression, we reach the following equation after rearranging:

$$L_0[J](p) = L_0[I](p) + [\overline{G}_{\sigma_p} * (I - I_p)f(M - M_p)](p). \tag{4}$$

We obtain the two-level joint local filter by upsampling the residual, adding it to both sides of the above formula, and expanding the convolution:

$$J_p = I_p + \sum_q \overline{G}_{\sigma_p}(q-p)f(M_q - M_p)(I_q - I_p), \tag{5}$$

where $q \in \Omega$ denotes the pixels in the local window Ω centered in the pixel p. The above formula shows that we can achieve joint filtering since the second term of the output J is a weighted average of the pixels in the spatial neighborhood using $\overline{G}_{\sigma_p}$ and the function f of the guidance image M, which has the similar spirit as the joint bilateral filter.

3.2 Unnormalized joint bilateral filter

In this section we discuss the relationship between the two-level joint local Laplacian filter and the original joint bilateral filter. The definition of the original joint bilateral filter (JBF) is expressed as:

$$\begin{aligned} JBF_p &= \frac{1}{W_p}\sum_q G_{\sigma_s}(q-p)G_{\sigma_r}(M_q - M_p)I_q, \\ W_p &= \sum_q G_{\sigma_s}(q-p)G_{\sigma_r}(M_q - M_p), \end{aligned} \tag{6}$$

where W_p is the normalization term and M is the guidance image, G_{σ_s} and G_{σ_r} denote the spatial and the range kernels, and they are typically Gaussian functions $G_\sigma(p) = exp(-p^2/2\sigma^2)$, respectively.

Considering the symmetry of the Gaussian kernel and the fact that the weights sum up to 1, the above formula can be rewritten as:

$$JBF_p = I_p + \frac{1}{W_p}\sum_q G_{\sigma_s}(q-p)G_{\sigma_r}(M_q - M_p)(I_q - I_p). \tag{7}$$

Comparing Eq. 5 and Eq. 7, we see that the two-level joint local Laplacian filter shares similarities with the original JBF. If we define $\overline{G}_{\sigma_p}$ as the spatial weight and f as G_{σ_r}, the only difference between the filters is that the weights are not normalized by $\frac{1}{W_p}$ in the two-level joint local Laplacian filter. This relationship leads to a new filter called the *unnormalized joint bilateral filter* by using $\sigma_p = \sigma_s$.

$$J_p = I_p + \sum_q \overline{G}_{\sigma_s}(q-p)G_{\sigma_r}(M_q - M_p)(I_q - I_p). \tag{8}$$

Figure 2 shows the difference between the unnormalized joint bilateral filter and the original joint bilateral filter. The filters have almost the same results within pure texture regions. The main difference between the two filters appears at strong edges. The original joint bilateral filter smooths some strong edges more aggressively than the unnormalized version. When a pixel p is significantly different from its neighbors, i.e., at strong edges, the normalization factor $\sum_q \overline{G}_{\sigma_s}(q-p)G_{\sigma_r}(M_q - M_p)$ is small and the output of the unnormalized joint bilateral filter is closer to the original input. That is, the unnormalized version performs better in terms of preserving structure edges of an image.

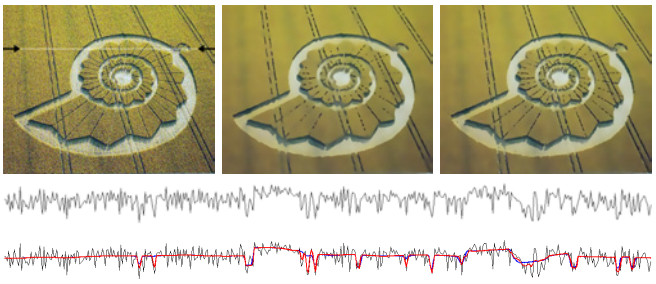


Fig. 2 Comparison of the unnormalized joint bilateral filter and the original joint bilateral filter with the same guidance image. From left to right of the top row: Input image, the result of the original joint bilateral filter, the result of the unnormalized version. The middle row shows the input 1D signal at the 94th line, the bottom row shows the corresponding filtering results. The blue signal denotes the result of the original joint bilateral filter, and the red signal is the result of the unnormalizedd version. Compared to the original joint bilateral filter, the unnormalized version may preserve structure edges better.

3.3 Guidance image

We now go back to the goal of texture smoothing. Akin to Subr et al. [21], we define textures as fine-scale spatial oscillations of signals. Cho et al. [5] showed that the joint bilateral filter can be effectively applied on structure-preserving texture smoothing by substituting a special texture description image, namely the guidance image, in the range kernel. To guide texture smoothing, the guidance image M should have similar values in a homogeneous texture region and distinguishable values across salient edges. That is, M should represent the approximate structures of the input image.

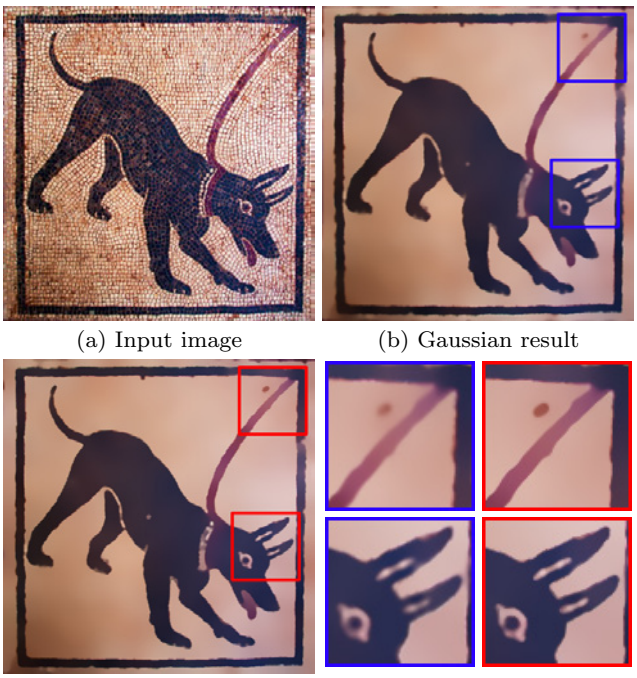
The Gaussian scale space theory has been developed by computer vision community to deal with structure identification in images with no prior information. We compute the guidance image M by two steps. Specifically, we first apply the Gaussian filter with the standard deviation σ_s to the input image I, i.e.,

$$D = I * G_{\sigma_s}. \tag{9}$$

After this step, details are eliminated. As a byproduct, however, the Gaussian filtering often blurs the large-scale structures of the image while eliminating the small-scale structures, and leads to blurring structure edges. Blurred results will be produced if we use D as the guidance image M directly, as shown in Fig. 4. In order to represent the clear structures of the image, we then employ the original joint bilateral filter with D

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

Fig. 3 Overview and intermediate images of our approach. From left to right: (a) Input image, (b) the initial blurred image D, (c) the final guidance image M, (d) the interpolation coefficient map μ, and (e) our filtering result J.



(a) Input image (b) Gaussian result

(c) Our result (d) Close ups

Fig. 4 The effect of the guidance image. (b) and (c) are the filtering results using guidance images D and M, respectively.

as the guidance image, to recover edges of the large-scale structures. We finally define our guidance image M as the output of the joint bilateral filter. That is, the guidance image M is defined as follows:

$$M = JBF(I, D), \tag{10}$$

where D is the result of the Gaussian filtering. Figure 3(c) shows the guidance image M via the joint bilateral filtering, which effectively restores structure edges from the initial blurred image D (Fig. 3(b)).

3.4 Texture smoothing via interpolation

By now, we have obtained our texture smoothing filter, i.e., the unnormalized joint bilateral filter, via the two-level joint local Laplacian filter. We will further prove that this new filter is equivalent to a spatially varying interpolation between the input image and the result of the joint bilateral filter. We find that this interpolation-based texture smoothing scheme is simple and intuitive.

We first define μ_p as $\sum_q \overline{G}_{\sigma_s}(q-p)G_{\sigma_r}(M_q - M_p)$. Since $\overline{G}_{\sigma_s} = \frac{1}{\sqrt{2\pi\sigma_s{}^2}}G_{\sigma_s}$ is a normalized Gaussian kernel and $\sum_q G_{\sigma_s}(q-p)G_{\sigma_r}(M_q-M_p)(I_q-I_p) = W_p(JBF_p - I_p)$ according to the expression of the original JBF, we can rewrite the formula of the unnormalized joint bilateral filter and substitute μ_p into the expression. Finally, we obtain the following formula:

$$\begin{aligned} J_p &= I_p + \sum_q \overline{G}_{\sigma_s}(q-p)G_{\sigma_r}(M_q - M_p)(I_q - I_p) \\ &= I_p + \frac{1}{\sqrt{2\pi\sigma_s{}^2}}\sum_q G_{\sigma_s}(q-p)G_{\sigma_r}(M_q - M_p)(I_q - I_p) \\ &= I_p + \frac{1}{\sqrt{2\pi\sigma_s{}^2}}W_p(JBF_p - I_p) \\ &= I_p + \mu_p(JBF_p - I_p) \\ &= (1-\mu_p)I_p + \mu_p JBF_p, \end{aligned} \tag{11}$$

where μ_p is the interpolation coefficient. In uniform regions of the guidance image M, the coefficient μ_p is large because $M_q - M_p$ is close to zero. On the contrary, in discontinuity regions of the guidance image, the coefficient μ_p is small around edges and this leads the unnormalized joint bilateral filter having a weaker filtering effect. This also explains why the unnormalized joint bilateral filter can effectively preserve the original edge shapes of the input.

In experiments, we find that a single iteration of the unnormalized joint bilateral filter may not be sufficient for obtaining a desired texture smoothing result. Therefore, we apply our unnormalized joint bilateral filter in an iterative manner resembling the method of Cho et al. [5]. Algorithm 1 depicts our final algorithm and Fig. 3 visualizes the flow of our approach with intermediate images.

(a) Input image (b) $\sigma_s = 3$ (c) $\sigma_s = 4$ (d) $\sigma_s = 5$ (e) $\sigma_s = 6$

Fig. 5 Different filtering results by progressively increasing the scale σ_s. In this example, we set $\sigma_r = 0.04$ and $N_{iter} = 5$.

Algorithm 1 Joint Local Laplacian Texture Filtering

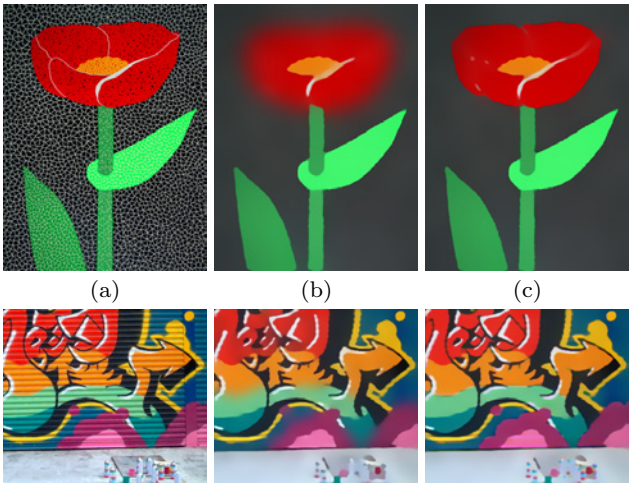
Input: Image I, N_{iter}, σ_s and σ_r
Output: Filtered image J

for $k = 1 \rightarrow N_{iter}$ **do**
  $D \leftarrow I * G_{\sigma_s}$ ▷ Gaussian blurring of I
  $M \leftarrow JointBilateralFiltering(I, D)$
  $\mu \leftarrow \sum_q \overline{G}_{\sigma_s}(q-p) G_{\sigma_r}(M_q - M_p)$
  $JBF \leftarrow JointBilateralFiltering(I, M, \sigma_s, \sigma_r)$
  $J \leftarrow (1-\mu)I + \mu JBF$
  $I \leftarrow J$
end for

4 Analysis

Parameters In our implementation, all pixel values are firstly normalized to the range $[0, 1]$ and the spatial kernel half-size of the joint bilateral filter is set to σ_s. Our algorithm can remove textures of different scales by varying the value of σ_s since the spatial standard deviation σ_s indicates the scale of textures to be suppressed (Fig. 5). Because there are no explicit measures to clearly distinguish scales of the main structures and textures of an image, the user specifies the value of σ_s directly to generate the final result. We find that satisfactory results can be achieved when $\sigma_s \in \{3, 4, 5, 6, 7\}$, $\sigma_r \in \{0.04, 0.05, 0.055\}$ and $N_{iter} \in \{5, 6, 7, 8, 9, 10\}$. Alternatively, we may also tweak the range standard deviation σ_r^k for the kth iteration by setting $\sigma_r^k = \sigma_r / \min(k, \lambda)$, where σ_r denotes the initial range standard deviation and λ is set to 3 or 4 in our experiments. This enables our algorithm to restore the original color variations of the object surface, as shown in Fig. 1.

Color image For a color image, we first convert it to grayscale and compute the guidance image M from this grayscale image. Then, our unnormalized joint bilateral filter is applied on each color channel using M to obtain the filtering result. We employ the contrast preserving decolorization algorithm [14], which aims to preserve the original color contrast as much as possible, to obtain grayscale images for most examples in this paper. Compared to simply computing a grayscale one via a linear combination of R, G, and B channels with fixed weight (e.g., the $rgb2gray()$ function in Matlab), using contrast preserving decolorization usually produces better filtering results although it adds some computation cost, as shown in the first row of Fig. 6.



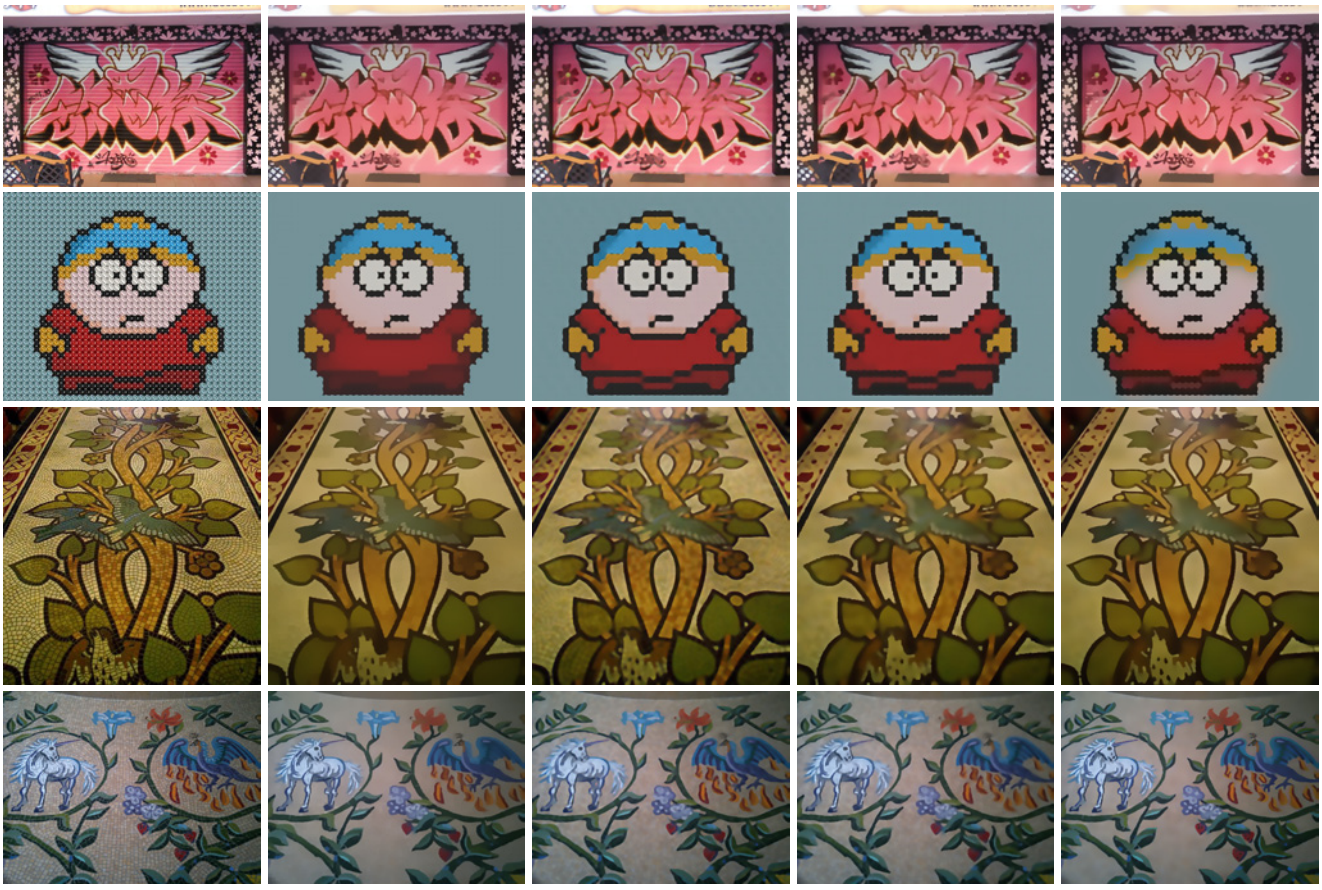
(a) (b) (c) (d) (e) (f)

Fig. 6 Color image filtering. (a) Input image. (b) The result using the $rgb2gray()$ function of MATLAB. (c) The result using contrast preserving decolorization [14]. (d) Input image. (e) The result using grayscale guidance image. (f) The result using color guidance image. Color guidance image preserves structures of an image better.

Color-to-gray conversion inevitably results in information loss. Alternatively, we may use full color information to compute the guidance image. We first apply the Gaussian filter and the joint bilateral filter in each channel of the color image to construct the color guidance image M. The interpolation coefficient of each channel is then computed via the corresponding channel of the guidance image M. Finally, we apply the joint bilateral filter by using the color M, and blend the input with the result of the joint bilateral filter using the interpolation coefficient to obtain the final result. The second row of Fig. 6 shows an example.

5 Results and Applications

In this section, we show the results produced by our algorithm and compare our approach with the state-



(a) Input images (b) Xu et al. [26] (c) Zhang et al. [28] (d) Cho et al. [5] (e) Our method

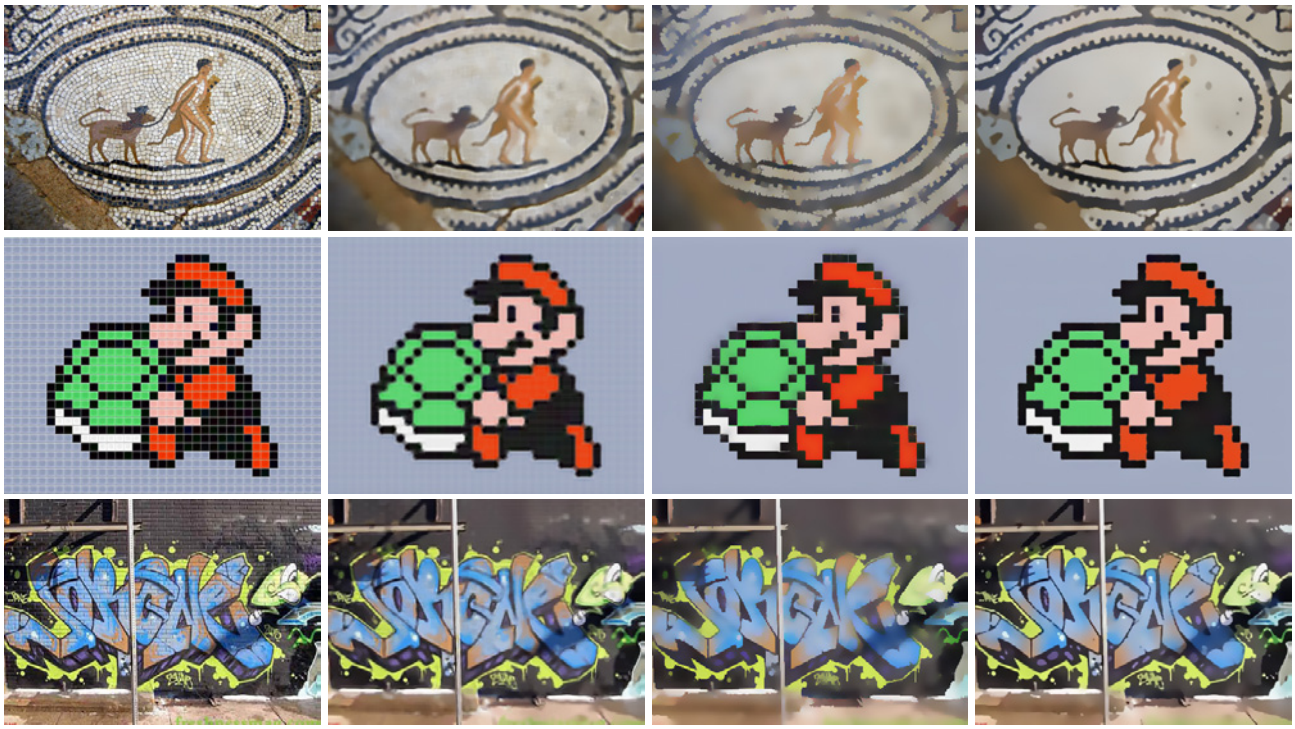
Fig. 7 Comparison with the state-of-the-art methods. The interpolation scheme ensures that our filter can effectively preserve the original edge shapes. Our method performs better in terms of preserving small objects while eliminating textures (the bottom row). The parameter values of the state-of-the-art methods are listed in the supplementary material.

of-the-art methods [3,5,11,26,28]. We carefully tuned the parameters to generate results of these methods. The experiments show that our method produces satisfactory filtering results and is comparable of previous works on structure-preserving texture smoothing.

Figure 1, Figure 7 and Figure 8 show comparisons between our method and previous works [3,5,11,26,28]. In Fig. 1, our method can effectively remove textures in both small and large regions. By gradually decreasing the values of σ_r, our method performs better than the method of Xu et al. [26] in terms of color variations. Compared to the method of Karacan et al. [11], our method effectively preserves structure edges, which are overblurred in their result. Our method can even preserve partial structures of the fish teeth, which are removed by Zhang et al. [28] and Cho et al. [5]. The method of Cho et al. [5] damages the original smoothness of structure edges and leads to jagged transitions. In contrast, our method can yield smooth edges and preserve the original edge shapes as much as possible.

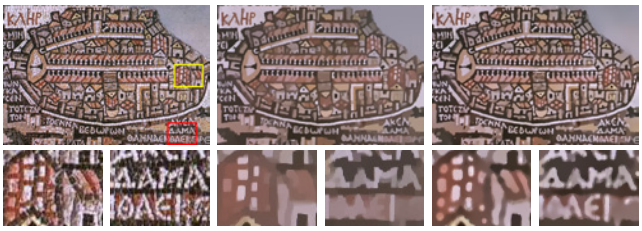
Figure 7 shows that both our method and Xu et al. [26]'s method can preserve the original structure edge shapes, which are altered in the results of Zhang et al. [28] and Cho et al. [5]. But with the method of Xu et al. [26] it is difficult to eliminate textures located near a structure edge, as pointed out in Cho et al. [5], and it may mistake parts of structures as textures (see Fig. 9). As shown in Fig. 8, Bao et al. [3]'s approach effectively removes textures in pure texture regions, but may remain textures located near prominent structure edges. The approach by Karacan et al. [11] may oversmooth structures while removing texture effectively. Our method can separate image structures and textures more cleanly. More results are supplied in the supplementary material.

Our experimental environment involves a computer with a CPU of Intel Core I5-4590, 4GB memory, and Matlab version 2013b. For a 400×300 grayscale image, it takes about 0.23 second for $\sigma_s = 5$ and $\sigma_r = 0.05$ to accomplish one single iteration process using the grayscale guidance image. The computation time of our



(a) Input images (b) Karacan et al. [11] (c) Bao et al. [3] (d) Our method

Fig. 8 Texture smoothed by Karacan et al. [11], Bao et al. [3] and our method. The parameter values of the state-of-the-art methods are listed in the supplementary material.



(a) Input image (b) Xu et al. [26] (c) Our method

Fig. 9 Extracting structures whose scales and appearance are similar to those of textures. Some structures are removed by Xu et al. [26].

Table 1 Performance of our method (in seconds) and parameter values.

Example	Size of input	σ_s	σ_r	N_{iter}	Times
Fig. 7 (row 1)	627×441	5	0.05	6	2.864
Fig. 7 (row 2)	400×324	4	0.04	11	5.565
Fig. 7 (row 3)	495×536	4	0.05	7	3.509
Fig. 7 (row 4)	900×675	5	0.05	5	8.903
Fig. 8 (row 1)	500×356	7	0.04	8	2.427
Fig. 8 (row 2)	500×406	5	0.055	8	2.798
Fig. 8 (row 3)	400×284	3	0.055	6	1.673

unoptimized implementation and experimental parameter values are shown in Table 1. Note that we construct the color guidance images to yield corresponding filtering results for the second and fourth rows of Fig. 7, which inevitably adds to the computation time. We believe that the performance of our method can be further improved using GPU acceleration or available acceleration methods of the bilateral filter.

Applications While our focus is on structure-preserving texture filtering in this work, we can apply our filter to a variety of image applications, including detail enhancement, edge detection, inverse halftoning, JPEG artifact removal, image abstraction, and image composition. We show some of them in this section. More application results are contained in our supplementary material.

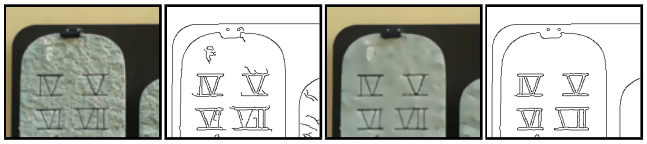
We can use our filter in layer decomposition and achieve detail enhancement results by enhancing the detail layer. A single iteration is sufficient for detail enhancement using our method. The interpolation scheme of our method yields better results with smooth edges than those of Cho et al. [5]. Fig. 10 shows an example. Our method can also be used to remove severe compression artifacts of cartoon JPEG images. Our restoration results are comparable to those of previous methods, as shown in Fig. 11. Due to the ability of removing details and textures, our method is helpful to find main edges of an image, as shown in Fig. 12. Directly using

(a) Input image (b) Cho et al. [5] (c) Our method

Fig. 10 Detail enhancement by Cho et al. [5] and our method. Parameters: Cho et al. [5] ($k = 3, N_{iter} = 5$), our method ($\sigma_s = 4, \sigma_r = 0.04$).

(a) (b) (c)

Fig. 11 Comparison of cartoon JPEG artifact removal. (a) Input image. (b) The result of Cho et al. [5]. (c) The result of our method.



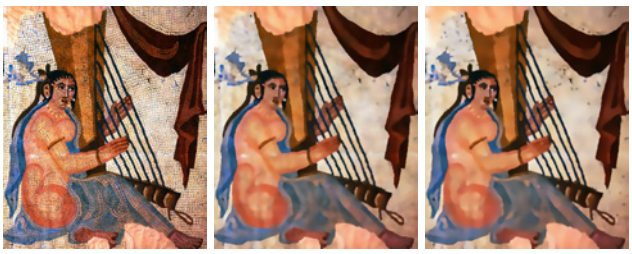
(a) (b) (c) (d)

Fig. 12 Edge detection by our method. (a) Input image. (b) Canny edge detection result of (a). (c) Our extracted structure component. (d) Our corresponding edge map. The threshold value for the Canny method in (b) and (d) is 0.38.

Canny edge detector on the input image cannot produce a cleaner structure edges map. Instead, our method produces the cleaner structure result, which makes edge detection easy to find structure edges.

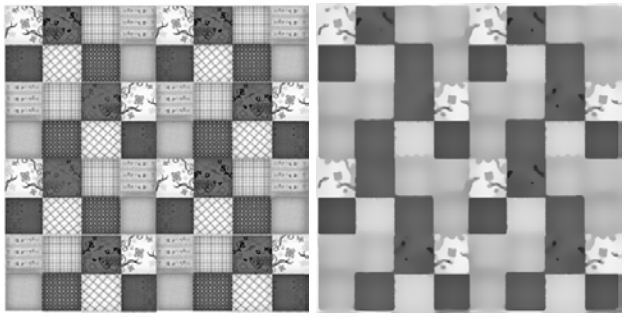
6 Discussion

The proposed method can be regarded as complementary to previous texture filtering approaches. From a formal standpoint, filtering results of our method are affected by the interpolation coefficient and the guidance image. Based on the contents of images, different filtering results can be achieved by setting different values of the interpolation coefficient and using different guidance images. If we set the interpolation coefficient μ_p to 1 for all images pixels, our method is the same as the original joint bilateral filter. Our method is allowed to accept various guidance images. For instance, other



(a) (b) (c)

Fig. 13 Using patch shift [5] as the guidance image. (a) Input image. (b) The result of Cho et al. [5]. (c) The result of our method using the guidance image of Cho et al. [5].



(a) Input image (b) Our result

Fig. 14 Failure example. Using a single scale leads to removing small-scale structures.

guidance image construction methods, such as patch shift [5], can also be employed here to better distinguish textures from structures. In this case, our method that produces smooth edges is a variant of Cho et al. [5]. Figure 13 shows such an example. The interpolation scheme plays a key role in distinguishing our approach from other methods. We believe that the applicability of the proposed unnormalized joint bilateral filter could be further explored in other extensive research on visual media processing.

Limitation The limitation is that our method uses a uniform scale to measure textures of an image, which may mistake some small-scale structures as textures and yield unsatisfactory texture filtering results. Figure 14 shows an example. A multiscale version of our approach might help to better measure texture features of different scales and improve filtering results.

7 Conclusion

We have presented a new filter for texture smoothing based on the two-level local Laplacian filter. By introducing a guidance image in the remapping function, the two-level joint local Laplacian filter owns the capability of structure-preserving texture smoothing while pre-

serving color variations. We further introduce the unnormalized joint bilateral texture filter that is closely related to the joint bilateral filter. Our method possesses better performance than the joint bilateral filter in terms of preserving structure edges while retaining the simplicity of the joint bilateral filter. Extensive experiments have been conducted to demonstrate the effectiveness of our method.

In the future, we plan to investigate a scale-adaptive texture smoothing method. This could improve the quality of structure-texture separation. Although using our current guidance image produces satisfactory results in most cases, we also try to develop more sophisticated texture measures to create the guidance image.

References

1. Mathieu Aubry, Sylvain Paris, Samuel W. Hasinoff, Jan Kautz, and Frédo Durand. Fast local laplacian filters: Theory and applications. *ACM Transactions on Graphics*, 33(5):167:1–167:14, 2014.
2. Jean-François Aujol, Guy Gilboa, Tony Chan, and Stanley Osher. Structure-texture image decomposition–modeling, algorithms, and parameter selection. *International Journal of Computer Vision*, 67(1):111–136, 2006.
3. Linchao Bao, Yibing Song, Qingxiong Yang, Hao Yuan, and Gang Wang. Tree filtering: Efficient structure-preserving smoothing with a minimum spanning tree. *IEEE Transactions on Image Processing*, 23(2):555–569, 2014.
4. Antoni Buades, Triet M. Le, Jean-Michel Morel, and Luminita A. Vese. Fast cartoon + texture image filters. *IEEE Transactions on Image Processing*, 19(8):1978–1986, 2010.
5. Hojin Cho, Hyunjoon Lee, Henry Kang, and Seungyong Lee. Bilateral texture filtering. *ACM Transactions on Graphics*, 33(4):128:1–128:8, 2014.
6. Antonio Criminisi, Toby Sharp, Carsten Rother, and Patrick P'erez. Geodesic image and video editing. *ACM Transactions on Graphics*, 29(5):134:1–134:15, 2010.
7. Zeev Farbman, Raanan Fattal, Dani Lischinski, and Richard Szeliski. Edge-preserving decompositions for multi-scale tone and detail manipulation. *ACM Transactions on Graphics*, 27(3):67:1–67:10, 2008.
8. Raanan Fattal. Edge-avoiding wavelets and their applications. *ACM Transactions on Graphics*, 28(3):1–10, 2009.
9. Eduardo S. L. Gastal and Manuel M. Oliveira. Domain transform for edge-aware image and video processing. *ACM Transactions on Graphics*, 30(4):69:1–69:12, 2011.
10. Kaiming He, Jian Sun, and Xiaoou Tang. Guided image filtering. In *Proceedings of the 11th European Conference on Computer Vision*, pages 1–14, 2010.
11. Levent Karacan, Erkut Erdem, and Aykut Erdem. Structure-preserving image smoothing via region covariances. *ACM Transactions on Graphics*, 32(6):176:1–176:11, 2013.
12. Michael Kass and Justin Solomon. Smoothed local histogram filters. *ACM Transactions on Graphics*, 29(4):100:1–100:10, 2010.
13. Johannes Kopf, Michael F. Cohen, Dani Lischinski, and Matt Uyttendaele. Joint bilateral upsampling. *ACM Transactions on Graphics*, 26(3), 2007.
14. Cewu Lu, Li Xu, and Jiaya Jia. Contrast preserving decolorization. In *IEEE International Conference on Computational Photography*, pages 1–7, 2012.
15. Ziyang Ma, Kaiming He, Yichen Wei, Jian Sun, and Enhua Wu. Constant time weighted median filtering for stereo matching and beyond. In *The IEEE International Conference on Computer Vision (ICCV)*, pages 49–56, 2013.
16. Sylvain Paris, Samuel W. Hasinoff, and Jan Kautz. Local laplacian filters: Edge-aware image processing with a laplacian pyramid. *ACM Transactions on Graphics*, 30(4):68:1–68:12, 2011.
17. P. Perona and J. Malik. Scale-space and edge detection using anisotropic diffusion. *IEEE Trans. Pattern Anal. Mach. Intell.*, 12(7):629–639, 1990.
18. Georg Petschnigg, Richard Szeliski, Maneesh Agrawala, Michael Cohen, Hugues Hoppe, and Kentaro Toyama. Digital photography with flash and no-flash image pairs. *ACM Transactions on Graphics*, 23(3):664–672, 2004.
19. Leonid I. Rudin, Stanley Osher, and Emad Fatemi. Nonlinear total variation based noise removal algorithms. *Phys. D*, 60(1-4):259–268, 1992.
20. Zhuo Su, Xiaonan Luo, Zhengjie Deng, Yun Liang, and Zhen Ji. Edge-preserving texture suppression filter based on joint filtering schemes. *IEEE Transactions on Multimedia*, 15(3):535–548, 2013.
21. Kartic Subr, Cyril Soler, and Frédo Durand. Edge-preserving multiscale image decomposition based on local extrema. *ACM Transactions on Graphics*, 28(5):147:1–147:9, 2009.
22. C. Tomasi and R. Manduchi. Bilateral filtering for gray and color images. In *Proceedings of the Sixth International Conference on Computer Vision*, page 839C846, 1998.
23. Joost van de Weijer and Rein van den Boomgaard. Local mode filtering. In *Computer Vision and Pattern Recognition (CVPR)*, pages 428–433, 2001.
24. Ben Weiss. Fast median and bilateral filtering. *ACM Transactions on Graphics*, 25(3):519–526, 2006.
25. Li Xu, Cewu Lu, Yi Xu, and Jiaya Jia. Image smoothing via l0 gradient minimization. *ACM Transactions on Graphics*, 30(6):174:1–174:12, 2011.
26. Li Xu, Qiong Yan, Yang Xia, and Jiaya Jia. Structure extraction from texture via relative total variation. *ACM Transactions on Graphics*, 31(6):139:1–139:10, 2012.
27. Wotao Yin, Donald Goldfarb, and Stanley Osher. Image cartoon-texture decomposition and feature selection using the total variation regularized l1 functional. In *Proceedings of the Third International Conference on Variational, Geometric, and Level Set Methods in Computer Vision*, pages 73–84, 2005.
28. Qi Zhang, Xiaoyong Shen, Li Xu, and Jiaya Jia. Rolling guidance filter. In *Computer Vision – ECCV 2014*, pages 815–830, 2014.
29. Qi Zhang, Li Xu, and Jiaya Jia. 100+ times faster weighted median filter. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2830–2837, 2014.