ORIGINAL ARTICLE

Edge guidance filtering for structure extraction

Beichen Sun[1] · **Yuehan Qi**[2] · **Guanyu Zhang**[1] · **Yang Liu**[1]

Abstract

Smoothing the multiscale, irregular, and high contrast textures while maintaining structures with small details is challenging for the existing texture filtering methods. In this paper, we put forward a novel edge guidance-based texture filter with an adaptive kernel scale scheme to address these challenges. The texture edges are identified by a texture edge detector first. Then, based on the texture edges, a variable per-pixel smoothing scale is selected to construct the scale map, which is used to guide the filtering. In the end, a novel pixel-selected filter is designed as post-processing to optimize the filtered images. The experimental results compared with the state-of-the-art methods show that our method has a better performance in suppressing different forms of textures while maintaining the main structure. In addition, our method can be applied well in a variety of image processing applications including: detail enhancement, inverse halftoning, virtual contour restoration and texture image segmentation.

Keywords Guided filtering · Texture filtering · Adaptive kernel scale scheme · Texture edges · Pixel-selected filter

1 Introduction

Texture filtering aims to remove textural details in an image while preserving structure edges as well as possible, which is essential and powerful operation in a variety of application such as detail enhancement [1], visual abstraction [2], image segmentation [3], edge detection [4], tone mapping [5], optical flow estimation [6] and illumination estimation [7]. The texture image is composed of structures and textures. Human vision can easily distinguish structures and textures of an image from a complex background. However, in the field of computer vision, extracting important structures from images with textures or complex backgrounds is still a challenging task.

In recent years, texture filtering algorithms have been broadly studied and some effective techniques are proposed, such as bilateral filter [8], guided filter [9], weighted least squares filter [10], iterative global optimization filter [2], edge selective joint filtering [11,12], Bayesian model averaging [13], patch geodesic paths filter [14], superpixel-guided smoothing [15], weighted guided image filter [16] and weighted median filter [17]. Although these filters have shown their excellent performance in many applications, most of these filters have no satisfactory performance on the following aspects. First, existing texture filtering algorithms are insufficient to maintain details such as slender structures and corners, as shown in Fig. 1a. Second, it is still a challenge on certain images to handle high-contrast and irregular texture patterns, as shown in Fig. 1b, c. A clear difference in gradient between texture and structure pixels is required. When the gradient of the texture is stronger than the gradient of the structure pixel, some gradient-based texture filtering methods will fail, i.e., a large number of textures with strong gradients will be preserved, and structures with small gradients will be suppressed. Besides, some methods can only handle periodically changing textures and have certain restrictions on texture types. Finally, for an image with

---

Beichen Sun and Yang Liu have contributed equally to this work.

✉ Yang Liu
   liu_yang@jlu.edu.cn

   Beichen Sun
   sunbc20@mails.jlu.edu.cn

   Yuehan Qi
   18yq36@queensu.ca

   Guanyu Zhang
   zhangguanyu@jlu.edu.cn

1   College of Instrumentation and Electrical Engineering, Jilin University, Ximinzhu Avenue, Changchun 130012, Jilin, China

2   Department of Art and Science, Queen's University, 99 University Avenue, Kingston, Ontario K7L 3N6,, Canada

multi-scale textures, as shown in Fig. 1d, the existing methods with a signal filter unavoidably lead to a certain degree of blurring effects in the image structures and texture residues for textures with large scales.

We address these challenges by constructing an edge guidance scale map to improve texture filtering. Texture filtering often involves image smoothing, which smooths out textures by averaging the pixel values around each pixel, while this indiscriminate filtering usually results in the loss of important structural information, such as structures. An effective solution to this problem is that the filtering scales of the pixels on the structures should be given small values to avoid blurring them out, while the filtering scales of the pixels on the textures should be given larger values to smooth out textures as much as possible.

In this paper, a texture filtering algorithm based on edge perception is proposed. First, texture edges are extracted by a texture edge detector, whose central idea is to use pixel-neighborhood statistics to distinguish structures from textures. Second, under the guidance of the texture edges, a scale map is constructed and then mean filtering within a circular neighborhood with a scale from the scale map is performed on the input image to get a coarse filtered image. Finally, a novel pixel-selected filter with a fixed-size window is used to optimize filtering results.

The rest of this paper is as follows. In Sect. 2, some other related works are reviewed briefly. In Sect. 3, the proposed filtering method is introduced, the scheme of texture edge extraction, the conception and construction of the scale map and the post-processing method. The performance of the new method is evaluated by carrying out various contrast experiments between the proposed method and state-of-the-art methods in Sect. 4, and we show the effects of the new method in some applications. Finally, we provide a summary in Sect. 5.

2 Related work

Many researchers have proposed many effective nonlinear image smoothing methods to solve image smoothing problems. These approaches are mainly considered as edge-preserving smoothing and structure-preserving smoothing [18,19]. Edge-preserving smoothing can retain the edge of image while filtering the image information. Structure-preserving smoothing aims to reduce the texture information and preserve the main-structure of image. Most structure-preserving smoothing methods are improved from edge-preserving filtering methods. Bilateral filtering (BLF) [20] is a classic average-based filtering method and it has been applied to a variety of computational photography applications. However, for multi-scale image decompositions, the method may lead to some halo artifacts due to the ongoing

coarsening process. Anisotropic diffusion model [21] preserves mainly image structures while eliminating noise and texture. Guided filter (GF) [9] can preserve the edge better rather than result in gradient reversals. In addition, many novel edge-preserving filters [22–24] have achieved better results.

State-of-the-art texture filtering methods can generally be classified into two categories, i.e., the traditional methods and learning-based methods. The traditional methods include local texture filtering methods and global texture filtering methods.

Local texture filtering is based on local features of the image, including average-based filtering and patch-based filtering. Karacan et al. [25] proposed a patch-based texture filtering method that uses a covariance matrix as a patch descriptor to distinguish local structures and textures, generating a better performance in distinguishing salient structures and texture. However, the method has the drawback that the performance is sensitive to the scale of the image feature. Zhang et al. [26] proposed the rolling guidance filter (RGF), which is based on the iteration of fast convergence to realize the rolling guided texture filtering. The implementation is simple and easy to understand. However, image structures will be blurred with the increasing of iteration numbers. By introducing a patch-shift mechanism to the bilateral filter, Cho et al. [5] proposed a bilateral texture filter (BTF), which significantly improves the texture filtering effect. However, the method may filter out small structures and the jagged edges are produced simultaneously. Lin et al. [27] made some improvements to the patch-offset method and improved joint bilateral filter to preserve small structures. Instead of patch shift, smaller patches are used to represent pixels located at structure edges, and original patches are still used to represent the texture regions. In addition, a windowed inherent variation is adapted to distinguish textures and structures for detecting structure edges. Although the method of [27] is slightly better than [25], there is no standard practice for threshold selection and only two fixed scales of the filter kernel are used in the method of [27]. In addition, there are some other local filtering methods proposed, such as adaptive texture filtering methods [28] and local-optimization filtering methods [29,30].

The global texture filtering methods are based on the idea of global optimization that seeks an optimal smoothed image by minimizing its difference from the input image and some smoothing constraints simultaneously, thus achieving the goal of suppressing the textures and highlighting the structures of the given image. Total variation (TV) regularization [31] can effectively remove textures while preserving large-scale edges but may filter out important structures. Farbman et al. [10] adopt a weighted least square-based optimization framework to smooth images. The L_0 image smoothing method [32] uses L_0 gradient as smoothing constraint and can

Fig. 1 Images with different-pattern textures. **a** Small details. **b** High contrast. **c** Irregular change. **d** Multiple scales

(a) small details   (b) high-contrast textures   (c) irregular textures   (d) multiscale textures
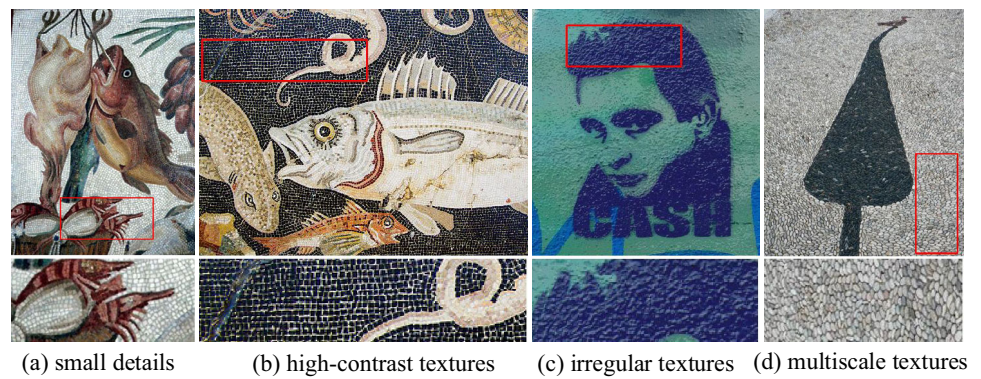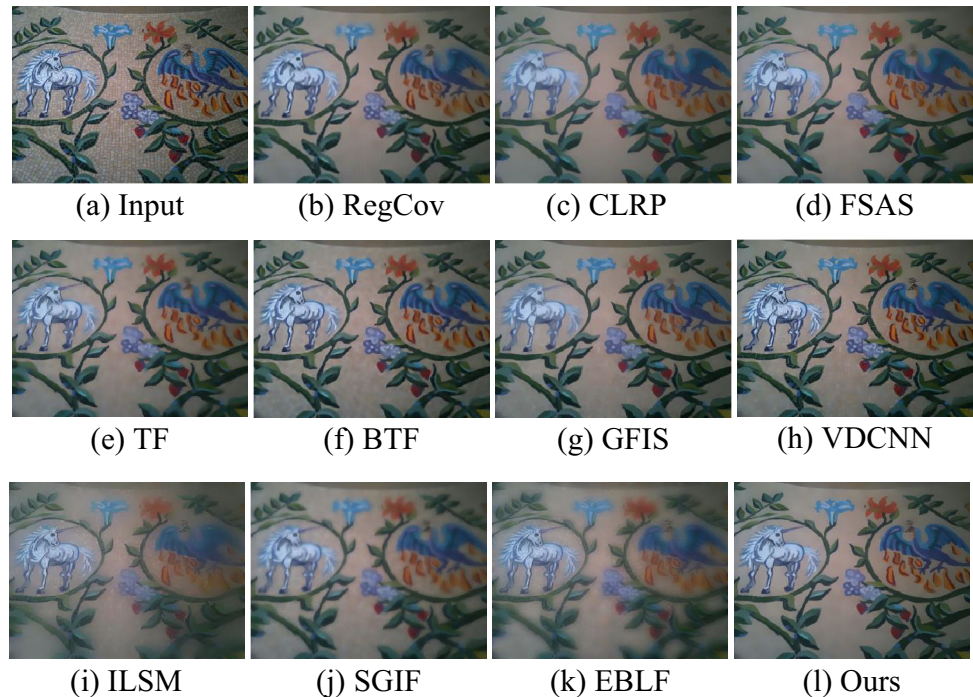
Fig. 2 Texture filtering results and comparison on the "Unicorn and Phoenix" mosaic image. Compared with previous methods, ours effectively filters out multiple-scale textures from the input image, while preserving structure edges and small-scale salient features, such as corners, without over-sharpening and over-blurring artifacts

(a) Input   (b) RegCov   (c) CLRP   (d) FSAS

(e) TF   (f) BTF   (g) GFIS   (h) VDCNN

(i) ILSM   (j) SGIF   (k) EBLF   (l) Ours

preferentially smooth pixels with a small gradient. The L_0 image smoothing method is simple in the model and performs well in texture removal. It has been widely used in image processing such as defogging and image enhancement. However, there are still problems in maintaining the image detail structure and removing large scale textures in the image. After further research, Xu et al. [4] proposed a related total variation (RTV)-based method, which is improved based on the TV model and considers the differences between textures and structures in terms of repeat patterns. The RTV method makes texture filtering effect greatly improved and can handle certain strong gradient textures. However, the parameter setting is an important factor for it to remove textures with strong gradients, even at the expense of blurring structure with weak gradients. Ham et al. [33] formulated the image filtering as a nonconvex minimization problem and designed an iterative smoothing way under the dynamic and static guidance, yet it is difficult to obtain its termination conditions. The above-

mentioned methods usually adopt one or two fixed filtering scales, leading to the difficulty in balancing the structure preserving and texture smoothing, and thus blur structures or remain details frequently emerging in their results. The ideal solution is that the texture region can have a larger filtering scale and smaller filter scales are used on structures and narrow corners to avoid fuzzy appearance.

Recently, convolutional neural networks (CNNs) have also shown great potential in edge-preserving filtering. Li et al. [34] proposed a CNN-based joint filter, whose architecture includes three sub-networks. The first two sub-networks are used to extract the features and then the features responses are concatenated as inputs for the third sub-network to reconstruct the filtered output. However, the dependencies modeled by them are quite implicit. In general, the neural network parameters of deep image smoothing operators should be trained under strong supervision with explicit training data. However, this explicit supervision would make the

smoothed images processed by deep operators only have one xed style. For this problem, Kim et al. [35] proposed a deep variational prior based on CNNs and combined it with an iterative smoothing process by using an alternating direction method of multiplier algorithm and its modular structure. There is no public and widely accepted dataset for an objective comparison of different edge-preserving smoothing algorithms, to solve this problem, Zhu et al. [36] proposed a benchmark by establishing a new dataset and a baseline model based on deep CNNs. A new weighted L_1 loss and weighted L_2 loss was introduced to train the baseline model on the dataset. The learning-based algorithmic often outperforms traditional methods by large margins, while the training based on large quantities of data still limits their works.

3 Proposed method

In this section, we will introduce the proposed edge guidance filtering, which mainly includes the following four parts: color quantization, texture edge extraction, scale map construction and pixel-selected filter, which are shown in Fig. 3. At first, the input image is processed by our proposed edge guidance filter, using the scale map as the scale guide image to create a coarse filtered image. Next, a novel pixel-selected filter is applied to the coarse filtered image, and the final filtered image is achieved at the given iteration.
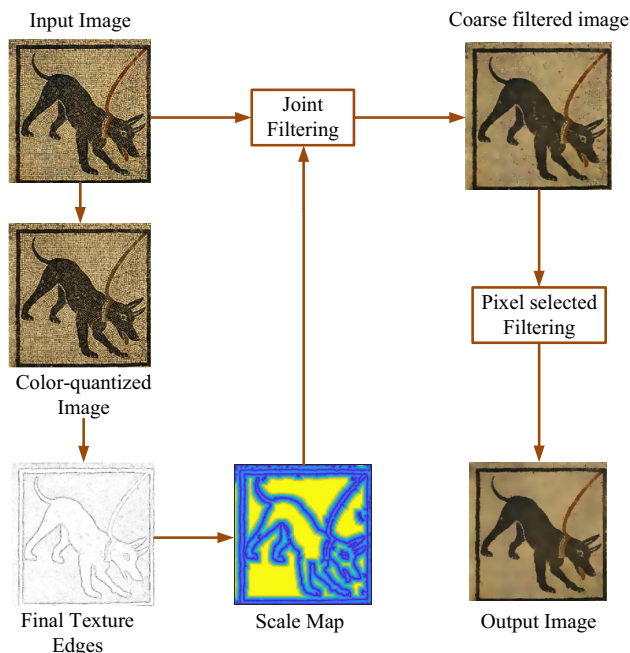

Fig. 3 Overall process and intermediate images of our edge guidance filtering

3.1 Color quantization

To extract the texture edge of the image, color quantization should be carried out first. Color quantization is a method of combining similar colors in an image into the same color. Color quantization can reduce the complexity of the image color, make the texture feature information of the image more explicit and make it possible to use the local histogram of the image to describe the texture feature. In this paper, mean-shift algorithm [37] is adopted to achieve color quantization of the image.

The mean-shift algorithm is an iterative process using the mean-shift vector. Given n sample points of R^d in d-dimensional space, for any point x in the space, the basic form of its mean-shift vector is defined as:

$$M_h = \frac{1}{J} \sum_{x_i \in S_J} (x_i - x) \tag{1}$$

where, $i = 1, \cdots, n$, S_J is a d-dimensional sphere with a radius h, J indicates that among these n sample points x_i, J points fall into the S_J region.

The process of color quantization using the mean-shift algorithm is as follows: In the 3-dimensional color space (note that the RGB and LAB color space can both be used here, but we prefer RGB color space since it can be directly used without color space conversion), choose K pixels as the centers and h as the radius to make K spheres. For every sphere, each pixel that falls inside the sphere will produce a vector, i.e., x_i, which starts at the center and ends at the pixel. K mean-shift vectors can be obtained from equation (1). Then using the end pixels of these mean-shift vectors as the centers make K new spheres and repeat the above steps to get mean-shift vectors. Repeated in this way, the mean-shift algorithm can converge to the colors with the K highest probability density, which is the color palette. Pixel mapping is performed by setting the colors of all pixels that fall in each sphere to the color of its center pixel.

The parameter K is the color number after color quantization and it decides the abilities to distinguish colors. As shown in Fig. 4, the more color bins, the stronger the ability to distinguish colors is. The image in Fig. 4 is a more extreme example, and $K = 20$ is adopted in most cases of our experiments, which is big enough to distinguish most common colors.

3.2 Texture edge extraction

We believe that the key to achieve structure extraction is to extract accurate structure edge information, that is, texture edge information of the image. Martin [38] provides an effective edge detection operator Pb detector, Pb detector can basically realize the edge detection of texture. In this

Fig. 4 Filtered results of our edge guidance filter with different color bins K. Colors with smaller difference can be distinguished as K increases

| Input | K=10 | K=20 | K=40 |
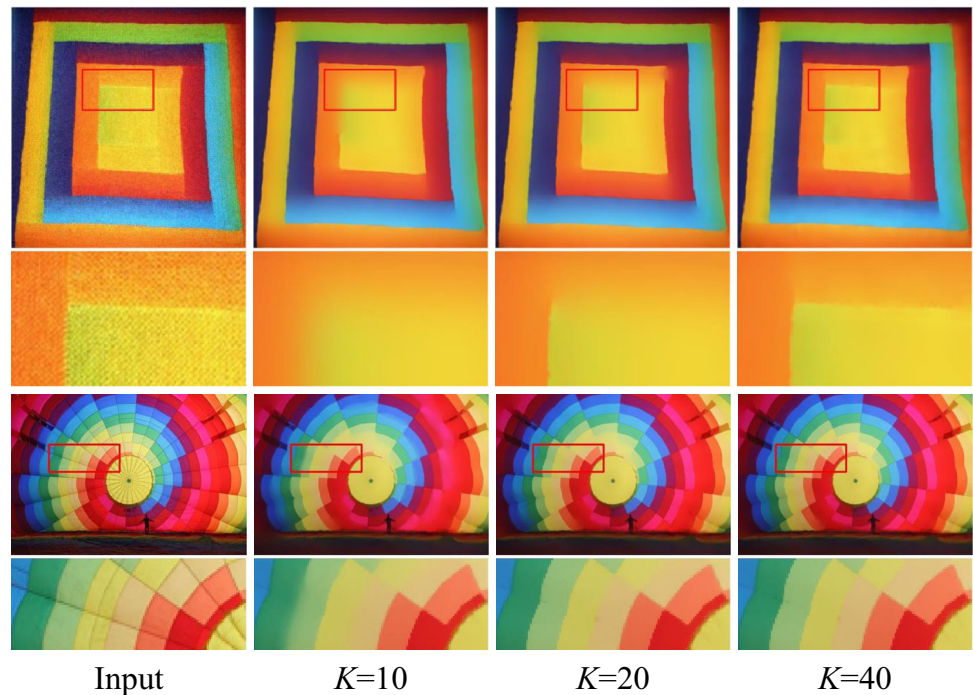
paper, we have improved the Pb detector to improve its effect on texture edge detection, and the improved Pb detector is described as follows.

Let p is a pixel on a color-quantized image where the color is divided into K bins. The circular region of pixel p with a radius of r is divided into two regions of equal size by a line passing through pixel p at an angle of θ. The texture descriptor of these two regions is represented by $g_{\theta,r}$ and $h_{\theta,r}$, respectively. $g_{\theta,r}$ and $h_{\theta,r}$ are two vectors used to represent color histograms within the regions. The variant of the Pb detector with arbitrary orientation is described as the histogram difference, which is computed by

$$G(p,r,\theta) = -ln\left(\sum_{k=1,\cdots,K}\sqrt{g_{\theta,r}(p,k)\,h_{\theta,r}(p,k)}\right) \quad (2)$$

where, k denotes the index of color bins.

Using Liu's method [39], we use eight equidistant angles for θ from 0° to 157.5°, and the texture edge detector mPb with multiple orientations θ and multiple scales r is expressed by

$$mPb(p,\theta) = \prod_{r\in\Omega}\sum_{\theta}G(p,r,\theta) \quad (3)$$

where $\Omega = \{r_1, r_2, \cdots, r_m\}$ is the set of radii. Different radii sets will lead to different filtering effects. As shown in Fig. 5, a set of bigger radii can ensure the filtering of large-sized textures.

mPb can be used to obtain the possibility of texture edge, and by combining with the gradient edge of the original image, accurate texture edge detection results can be obtained. The extraction process is shown in Fig. 6.

3.3 Scale map construction and multiscale filtering

For a given image I, it can be decomposed into two parts, i.e., the structure S and texture T, which can be expressed by

$$I = S + T \quad (4)$$

In this paper, we use circular mean filter to compute the structural component of pixel p as:

$$S(p) = \frac{\sum_{q\in C(p,\rho)} u_{pq} I(q)}{\sum_{q\in C(p,\rho)} u_{pq}} \quad (5)$$

where, $C(p,\rho)$ denotes a circular neighborhood with radius ρ centered at pixel p, pixel q is within $C(p,\rho)$ and its coordinates are given by $(x_q - x_p)^2 + (y_q - y_p)^2 \leq \rho^2$, u_{pq} denotes the location relationship between pixel p and pixel q in the circular neighborhood $C(p,\rho)$, and its values can be expressed by

$$u_{pq} = \begin{cases} 1, & (x_q - x_p)^2 + (y_q - y_p)^2 < \rho \\ s, & (x_q - x_p)^2 + (y_q - y_p)^2 = \rho \\ 0, & (x_q - x_p)^2 + (y_q - y_p)^2 > \rho \end{cases} \quad (6)$$

Fig. 5 Filtered results of our edge guidance filter with different radii sets Ω. Textures with bigger scales are removed as r_m increases
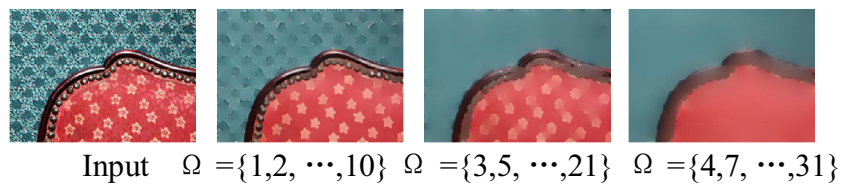
Input   Ω $=\{1,2,\cdots,10\}$ Ω $=\{3,5,\cdots,21\}$ Ω $=\{4,7,\cdots,31\}$

Fig. 6 Extraction process and intermediate images of texture edges

Input Image   Gradient Edges

Color-quantized Image   Initial Texture Edges   Hadamard Product   Final Texture Edges
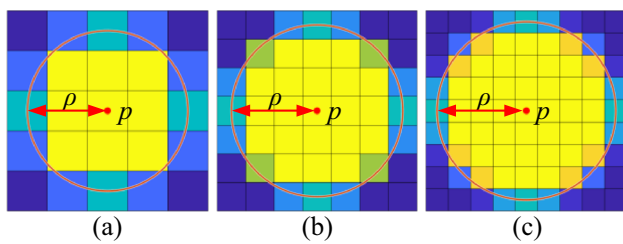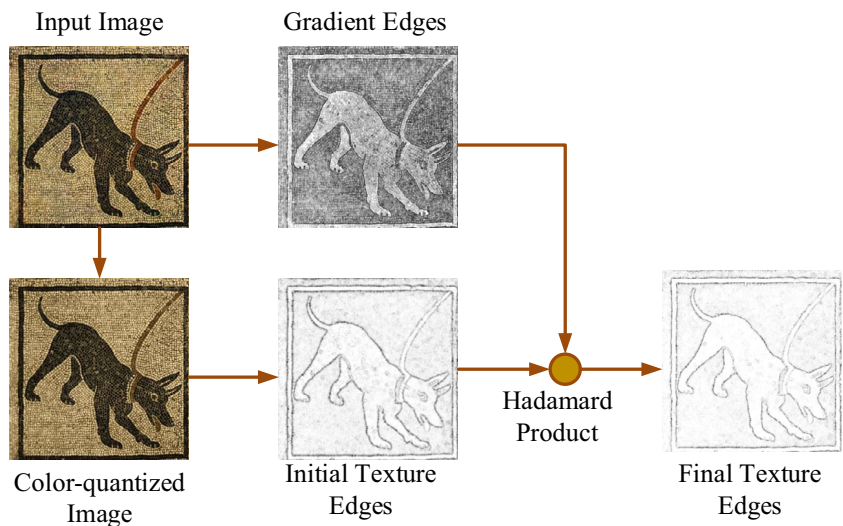
Fig. 7 Circular neighborhoods with different radius. **a** $\rho = 2$, **b** $\rho = 3$, and **c** $\rho = 4$
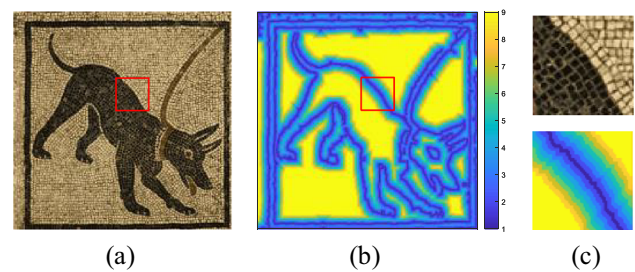
(a)   (b)   (c)

Fig. 8 Diagram of scale map. **a** Input image. **b** Scale map. **c** Close-ups

(a)   (b)   (c)

where, s denotes the value of the pixel not completely falling into $C(p, \rho)$. The pixel's u_{pq} is equal to the area within $C(p, \rho)$. The radius ρ is the filtering scale of the circular mean filter. Figure 7 shows examples of circular neighborhoods with different radii.

In order to filter out textures as much as possible while preserving the image structure, filters with different scales are needed. Intuitively, for the texture edge part of the image, a small-scale filter is needed to avoid blurring structures, while for the texture part, a large-scale filter is needed to smooth out the texture. Based on the texture edges extracted by the mPb detector, we now can construct the scale map by setting the filtering radii. We design a set of filtering radii $\Psi = \{R_1, \cdots, R_n\}$ to control the texture scale to be removed. Every radius is equally spaced in Ψ and subscript n denotes the number of radii, which is set to a fixed value of 10. R_1 and R_n is the minimum and maximum value, respectively. As shown in Fig. 8c, for pixels right on texture edges, the

filtering radii are set to the minimum radius R_1. For pixels near texture edges, the filtering radii are set to R_2, and so on, until the farthest pixels have the maximum radius R_n. That is to say, the closer a pixel is to the texture edge, the smaller the filtering radius is. When a pixel is far away from each texture edge, the filtering radius is set to R_n. In addition, for pixels in the texture region, the filtering radii are also set to R_n.

The parameter Ψ affects the filtered results of our method. R_1 affects the pixels near texture edges, and a smaller R_1 introduces less edge blurring. R_n controls the maximum scale of textures to be removed, a bigger R_n leads to more aggressive texture removal and smoother results. In this paper, we set $\Psi = \{1, 2, \cdots, 10\}$, which is suitable for images with sizes of around 480×320 pixels. For images with high resolutions, such as HD, 2K and 4K, we can set $R_1 = 1$ and the value of n in Ψ to be proportional to the image resolution. In

addition, the image resolution also affects the setting of Ω in Eq. (3), whose value also should increase proportionally to the image resolution.

3.4 Pixel-selected filter

Through mean filtering guided by scale map, the textures with large scales or strong gradients can be removed. However, there are still some residual textures near texture edges, which are shown as the coarse filtered image in Fig. 3. In order to perfect the filtering effect, we propose a novel modified post-processing method based on Liu's method [40], which is essentially a pixel-selected mean filter.

In natural pictures, the color distribution in the LAB color space has a better separation degree than in the RGB space, so we convert the image to the LAB color space for post-processing. The proposed pixel-selected filter is essentially a self-guided mean filter, and the filtering result of pixel p is computed by

$$S'(p) = \frac{\sum_{q \in N(p,w)} w_{pq} I(q)}{\sum_{q \in N(p,w)} w_{pq}} \tag{7}$$

where, $N(p, w)$ is a squared patch around centered pixel p with w pixels on its side, $I(q)$ is the value of pixel q in the input image, and w_{pq} denotes the weight of pixel q in the filter.

$$w_{pq} = \frac{1}{1 + e^{d_{pq} - d_0}} \tag{8}$$

where d_{pq} is the $L2$ norm between pixel p and pixel q in the LAB color space, and d_0 is a threshold.

As shown in Fig. 3, the result of the pixel-selected filter is assigned to the input image for the next iteration. Finally, the smoothed image is achieved at the given iteration.

For the pixel-selected filter, compared to BLF, it is a simpler, faster and effective edge preserve filtering method, there are three parameters, i.e., the patch scale w, distance threshold d_0 and iteration time n_{itr}. Patch scale w determines how large the scale of textures is to be smoothed. The larger the patch scale is, the larger the textures are removed while more structures could be damaged. However, if the patch scale is too small, the textures cannot be smoothed enough. Since textures with large scales have filtered out by our edge guidance filter, the patch scale set to 3 is suffice for most cases. The pixel-selected filter is adjusted by the other two parameters: d_0 and n_{itr}, whose results are shown in Fig. 9.

Observation on the filtered results from left to right shows the role of the n_{itr}. As n_{itr} increases, the texture region will be more flattened. While some noise with a bigger patch cannot be filtered out. This problem can be solved by increasing parameter d_0. The filtered results from top to bottom show the
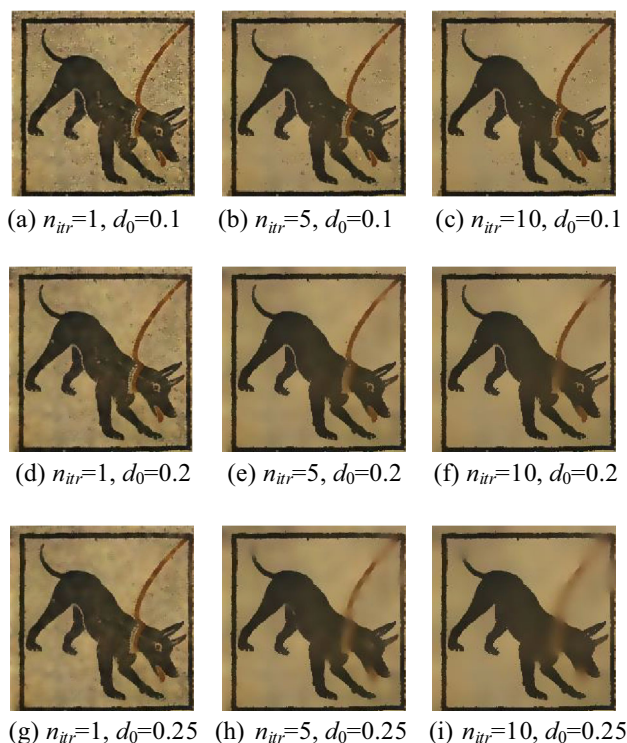

(a) n_{itr}=1, d_0=0.1   (b) n_{itr}=5, d_0=0.1   (c) n_{itr}=10, d_0=0.1

(d) n_{itr}=1, d_0=0.2   (e) n_{itr}=5, d_0=0.2   (f) n_{itr}=10, d_0=0.2

(g) n_{itr}=1, d_0=0.25   (h) n_{itr}=5, d_0=0.25   (i) n_{itr}=10, d_0=0.25

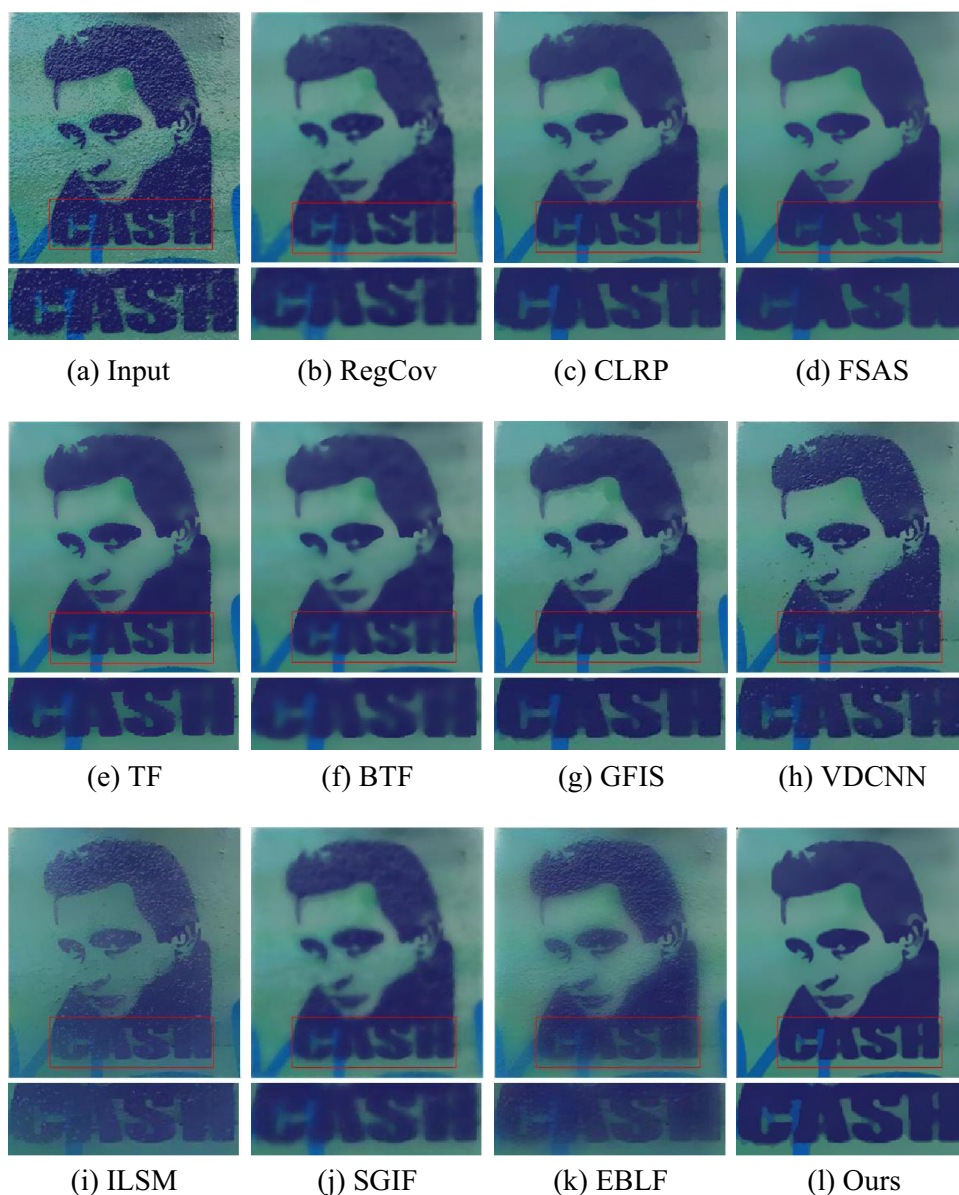**Fig. 9** Filtered results of our pixel-selected filter with varied parameters

effect of the d_0. As d_0 increases, more noise can be smoothed while the texture edges are blurred inevitable. In practice, we give d_0 a value from the set {0.1, 0.2, 0.3} and n_{tir} a value from the set {3, 5, 7, 9}, which can often produce the desired results, according to specific images and applications.

4 Experiments and applications

4.1 Experiments and evaluations

To evaluate the performance of the new method, a series of contrast experiments were conducted with the state-of-the-art texture filtering methods, including region covariance filter (RegCov) [25], fast scale-adaptive bilateral texture smoothing (FSAS) [8], customized low-rank prior model filter (CLRP) [41], tree filtering (TF) [29], bilateral texture filtering (BTF) [5], generalized framework for edge-preserving and structure-preserving image smoothing (GFIS) [18,19], real-time image smoothing via iterative least squares (ILSM) [22], semi-global weighted least squares in image filtering (SGIF) [23], embedding bilateral filter in least squares for efficient edge-preserving image smoothing (EBLF) [24] and CNN-based method (VDCNN) [36]. The parameters of all the methods being compared in the experiments are provided by the authors in their papers. Our method has

Fig. 10 Visual comparisons with the state-of-the-art texture filtering methods. **a** Input image, **b** RegCov [$k = 19$, $\sigma = 0.2$, $Model1$], **c** CLRP [$\gamma = 1.6$, $\beta = 1e-5$, $n_{iter} = 200$, $ModelB$], **d** FSAS [$\lambda = 4$, $k = 30$, $n_{iter} = 2$], **e** TF [$\sigma = 0.01$, $\sigma_s = 3$], **f** BTF [$k = 2$, $n_{iter} = 5$], **g** GFIS [$\lambda = 0.5$, $n_{iter} = 10$], **h** VDCNN [$layers = 20$ $k = 3$], **i** ILSM[$n_{iter} = 10$, $\lambda = 1$], **j** SGIF [$n_{iter} = 5$, $\sigma_R = 2$], **k** EBLF [$\sigma_s = 15$, $\sigma_r = 0.05$], and **l** Ours [$\Omega = \{1, 2, \ldots, 20\}$, $d_0 = 0.13$, $n_{itr} = 5$]
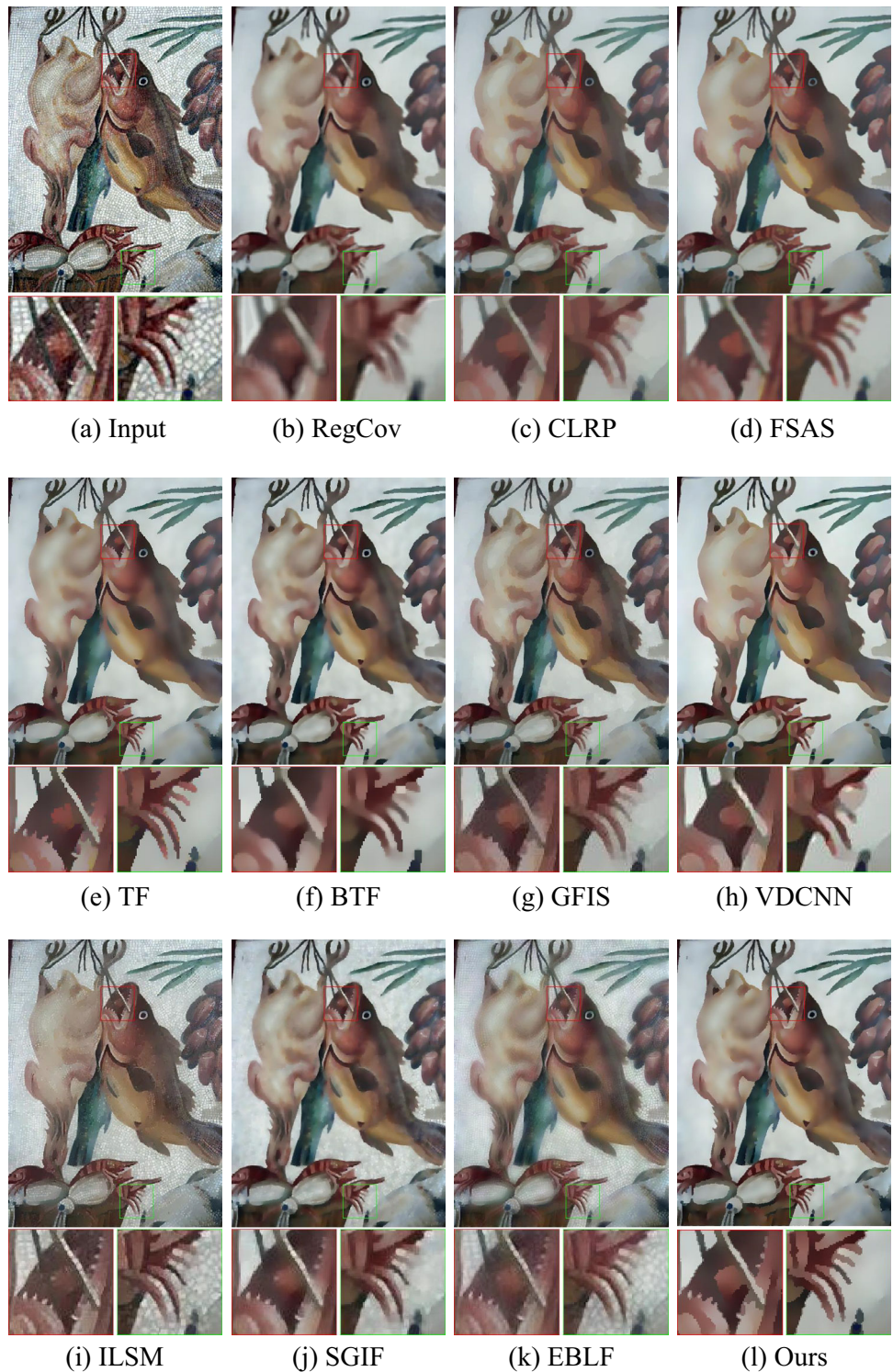
(a) Input	(b) RegCov	(c) CLRP	(d) FSAS
(e) TF	(f) BTF	(g) GFIS	(h) VDCNN
(i) ILSM	(j) SGIF	(k) EBLF	(l) Ours

four parameters that need to be adjusted: the color bin number K mentioned in Sect. 3.1, the radii set Ω mentioned in Sect. 3.2, the distance threshold d_0 mentioned in Sect. 3.4 and the iteration time n_{itr} mentioned in Sect. 3.4. When our method carried out the experiment in this paper, K values are all 20, and the settings of other parameters are described in the experimental figure notes. The code is publicly available at https://github.com/LeonSunn/Edge-Guidance-Filtering-For-Structure-Extraction.

We compare with three excellent edge-preserving filtering approaches including ILSM, SGIF and EBLF. These methods are effective in general texture filtering. However, experiments show that it is difficult to smooth high contrast texture only by edge-preserving filters. These algorithms will not be discussed in more detail in subsequent sections.

Figure 10 shows eleven results on a texture image whose background rock particles are textures and the graffiti is the structure, including a portrait and letters. As shown in Fig. 10, all the methods in the texture removal test can extract prominent image structure while filtering out the textures. However, we notice that the methods of RegCov, CLRP and VDCNN perform less well in removing background textures because these backgrounds of filtered images are not flat. Moreover, the methods FSAS, TF, BTF and GFIS over-blur the structural edges, and the letters CASH in the filtered graffiti images are seriously damaged and illegible. Although the TF method overcomes these shortcomings, there is another problem. The TF method tries to find the texture regions with a minimum spanning tree to avoid using axis-aligned windows. While this pixel-based method may mistake the region to which

Fig. 11 Visual comparisons with different methods on preserving small structure edges. **a** Input image, **b** RegCov $[k = 15, \sigma = 0.2, Model1]$, **c** CLRP $[\gamma = 1.6, \beta = 1e - 5, n_{iter} = 200, ModelB]$, **d** FSAS $[\lambda = 4, k = 30, n_{iter} = 2]$, **e** TF $[\sigma = 0.01, \sigma_s = 3]$, **f** BTF $[k = 2, n_{iter} = 5]$, **g** GFIS $[\lambda = 0.7, n_{iter} = 10]$, **h** VDCNN $[layers = 20\ k = 3]$, **i** ILSM $[n_{iter} = 9, \lambda = 0.9]$, **j** SGIF $[n_{iter} = 12, \sigma_R = 2.5]$, **k** EBLF $[\sigma_s = 7, \sigma_r = 0.07]$, and **l** Ours $[\Omega = \{1, 2, \ldots, 20\}, d_0 = 0.13, n_{itr} = 5]$

(a) Input   (b) RegCov   (c) CLRP   (d) FSAS

(e) TF   (f) BTF   (g) GFIS   (h) VDCNN

(i) ILSM   (j) SGIF   (k) EBLF   (l) Ours

Fig. 12 Results and comparison on "Pompeii Fish Mosaic". **a** Input image, **b** RegCov [$k = 19$, $\sigma = 0.2$, $Model1$], **c** CLRP [$\gamma = 1.6$, $\beta = 1e - 5$, $n_{iter} = 200$, $ModelB$], **d** FSAS [$\lambda = 4$, $k = 30$, $n_{iter} = 2$], **e** TF [$\sigma = 0.02$, $\sigma_s = 3$], **f** BTF [$k = 2$, $n_{iter} = 5$], **g** GFIS [$\lambda = 0.7$, $n_{iter} = 10$], **h** VDCNN [$layers = 20\ k = 3$], **i** ILSM [$n_{iter} = 40$, $\lambda = 1$], **j** SGIF [$n_{iter} = 10$, $\sigma_R = 2$], **k** EBLF [$\sigma_s = 12$, $\sigma_r = 0.15$], and **l** Ours [$\Omega = \{1, 2, \ldots, 20\}$, $d_0 = 0.13$, $n_{itr} = 5$]

(a) Input (b) RegCov (c) CLRP (d) FSAS

(e) TF (f) BTF (g) GFIS (h) VDCNN

(i) ILSM (j) SGIF (k) EBLF (l) Ours

some pixels belong, thus producing loose edges, as shown in Fig. 10e. By comparison, our method totally removes rock particles while preserving the graffiti without blurring it, i.e., the portrait and letters in the filtered images are clear and recognizable, as shown in Fig. 10l.
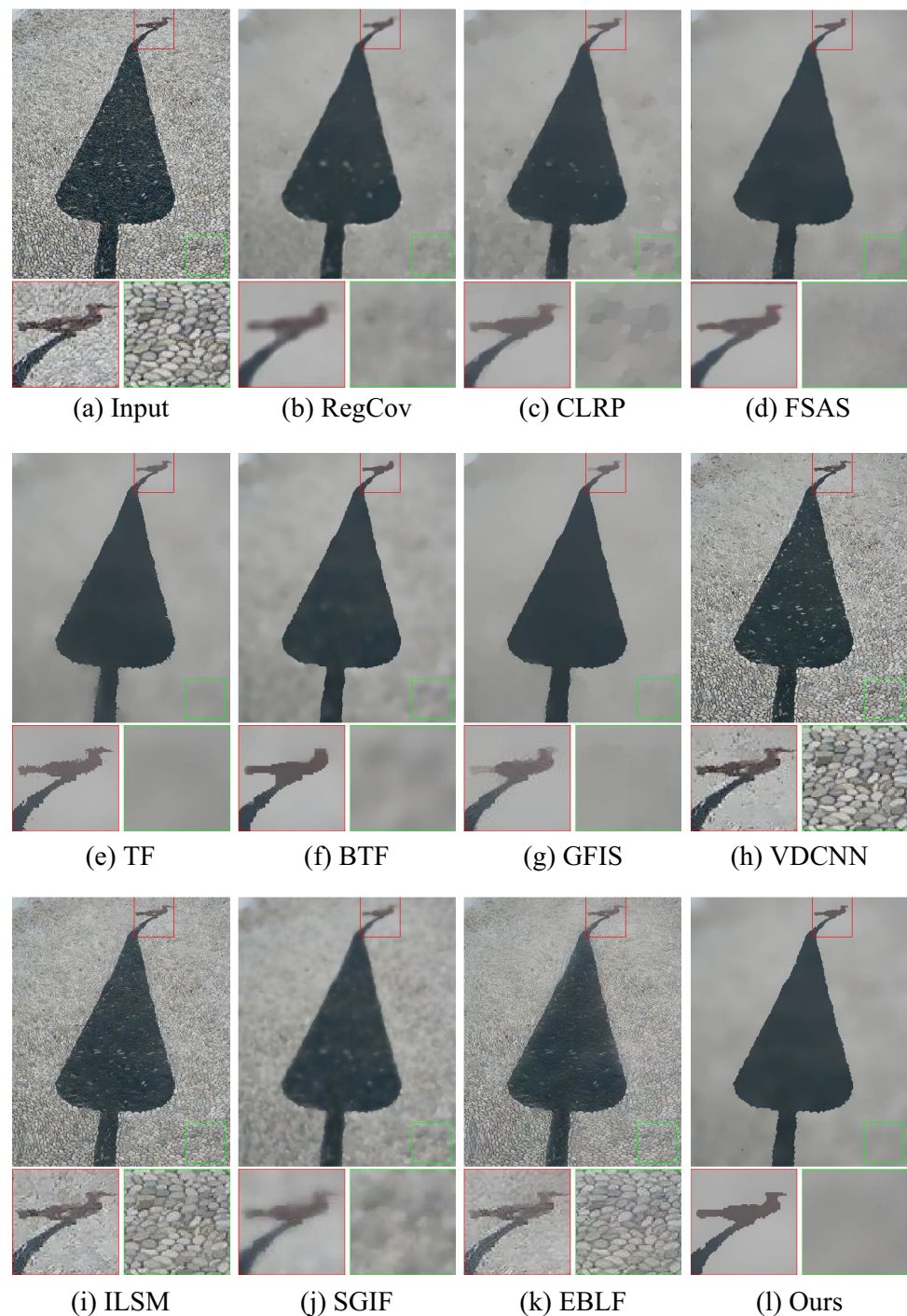
To further demonstrate the effectiveness of our method in preserving small details, Fig. 11 shows the texture removal results of different methods on a mosaic image, which contains textures and a large number of small structures, such as the teeth of fish, and the details of shrimp. Although all the methods can extract prominent image structures, there are some differences in the ability to preserve small structures. For example, as shown in the enlarged box in Fig. 11b, f, h, the methods of RegCov, BTF and VDCNN may blur small structure edges, such as the details of shrimp, and they smooth out some small structures, such as the teeth of fish. In Fig. 11c, d, e, g, the methods of CLRP , FSAS, TF and GFIS perform well in removing textures and preserving small structures. Among all these methods, our method performs

best in preserving small structures, i.e., only our method can both preserve all the teeth of fish and the details of shrimp, as shown in the enlarged box in Fig. 11l.

Figure 12 shows an ancient mosaic image from Pompeii, which contains complex structures with weak gradients and textures with strong gradients. As shown in Fig. 12b, c, e, the methods of RegCov, CLRP and TF cannot effectively filter out high-contrast mosaic textures in the background. The methods of RegCov, TF, BTF, GFIS and VDCNN have problems of preserving fine and complex structures, such as the big fish's eyes and its surroundings as shown in the left enlarged boxes in Fig. 12b, e, f, g, h. The methods of RegCov, TF and BTF may over-blur structural detailed edges, such as the red stripes shown in the right enlarged boxes in Fig. 12b, e, f.

Figure 13a shows an image of a cobblestone ground with different texture scales at different locations due to perspective. Since existing texture filtering methods usually adopt fixed-scale kernels for structure-texture separation, it

Fig. 13 Results and comparison on the image with multiscale textures. **a** Input image, **b** RegCov [$k = 19$, $\sigma = 0.2$, $Model\,1$], **c** CLRP [$\gamma = 1.6$, $\beta = 1e - 5$, $n_{iter} = 200$, $Model\,B$], **d** FSAS [$\lambda = 4$, $k = 30$, $n_{iter} = 2$], **e** TF [$\sigma = 0.08$, $\sigma_s = 3$], **f** BTF [$k = 3$, $n_{iter} = 5$], **g** GFIS [$\lambda = 1.5$, $n_{iter} = 15$], **h** VDCNN [$layers = 20\ k = 3$], **i** ILSM [$n_{iter} = 35$, $\lambda = 1$], **j** SGIF [$n_{iter} = 30$, $\sigma_R = 1.9$], **k** EBLF [$\sigma_s = 15$, $\sigma_r = 0.15$], and **l** Ours [$\Omega = \{1, 2, \ldots, 20\}$, $d_0 = 0.1$, $n_{itr} = 5$]
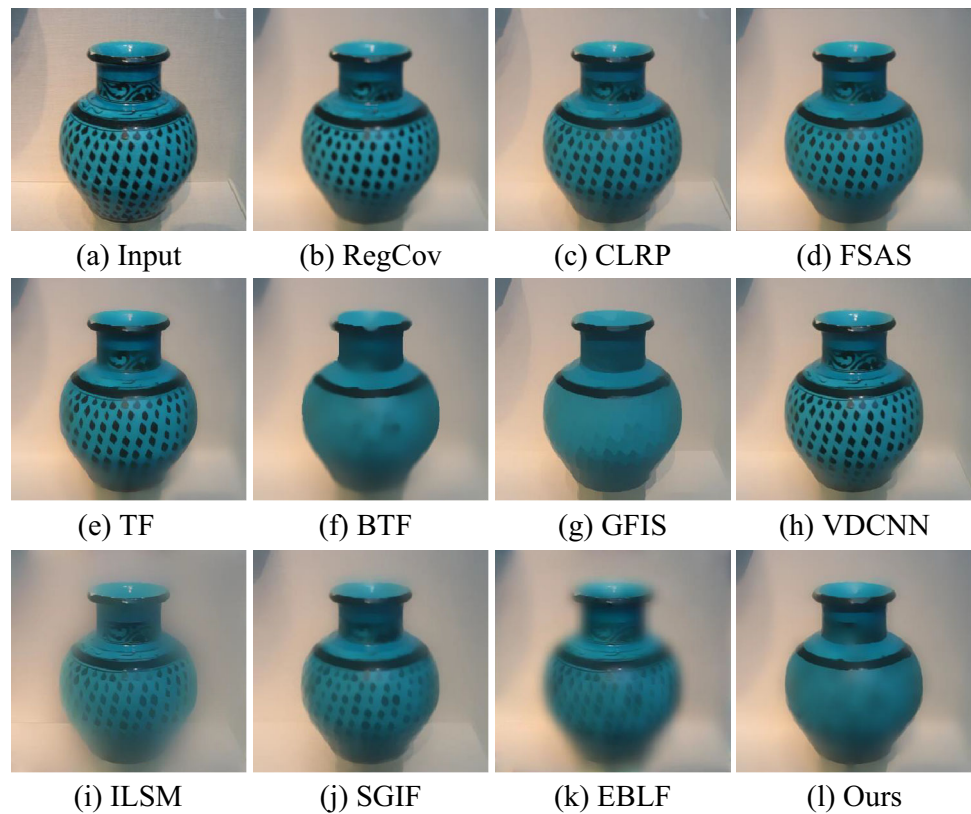

(a) Input (b) RegCov (c) CLRP (d) FSAS

(e) TF (f) BTF (g) GFIS (h) VDCNN

(i) ILSM (j) SGIF (k) EBLF (l) Ours

is a challenge for them to preserve salient but small-scale structures when removing large-scale textures shown in the enlarged boxes in Fig. 13b–h. The performances on suppressing the large-scale textures are unsatisfactory and the bird-like structure is destroyed to a varying extent. In contrast, as shown in Fig. 13l, since our method automatically identifies the smoothing scale per pixel, small but salient structures (bird-like structures) are well preserved even when large textures are smoothed out. This experiment demonstrates that our method outperforms other methods in filtering out multiscale textures.

It is still a challenge for existing methods to filter out large-scale textures with extreme variations while preserving important structures, which are shown in Fig. 14. Since our texture filter adopts adaptive kernel scales for structure-texture separation, our method outperforms the other methods in this regard.

Fig. 14 Results and comparison on the image with large scale textures. **a** Input image, **b** RegCov [$k = 19, \sigma = 0.2$, $Model1$], **c** CLRP [$\gamma = 1.6$, $\beta = 1e - 5, n_{iter} = 200$, $Model B$], **d** FSAS [$\lambda = 4$, $k = 30, n_{iter} = 2$], **e** TF [$\sigma = 0.15, \sigma_s = 5$], **f** BTF [$k = 6, n_{iter} = 7$], **g** GFIS [$\lambda = 2.5, n_{iter} = 20$], **h** VDCNN [$layers = 20\ k = 3$], **i** ILSM [$n_{iter} = 10, \lambda = 1$], **j** SGIF [$n_{iter} = 10, \sigma_R = 2$], **k** EBLF [$\sigma_s = 15, \sigma_r = 0.05$], and **l** Ours [$\Omega = \{3, 5, \ldots, 21\}$, $d_0 = 0.21, n_{itr} = 10$]

(a) Input   (b) RegCov   (c) CLRP   (d) FSAS

(e) TF   (f) BTF   (g) GFIS   (h) VDCNN

(i) ILSM   (j) SGIF   (k) EBLF   (l) Ours

Table 1 Quantitative comparison on the dataset. The best results are in **bold**. The second results are underlined

Method	RegCov	CLRP	FSAS	TF	BTF	GFIS	VDCNN	ILSM	SGTF	EBLF	Ours
PSNR	23.346	23.889	22.581	26.173	22.267	24.262	**30.225**	24.591	23.275	23.388	<u>26.320</u>
SSIM	0.8323	0.8569	0.8182	0.9057	0.8050	0.8734	**0.9663**	0.9098	0.8346	0.8761	<u>0.9116</u>

Table 2 Ablation study of image smoothing effects with no texture edges, no scalemap and no pixel. The best results are in bold

	PSNR	SSIM
No texture edges	24.850	0.905
No scalemap	22.575	0.839
No pixel selected	24.699	0.895
All module	**25.275**	**0.914**

We compare the average PSNR and SSIM [42] of Zhu's dataset [36] to evaluate our method, the objective performances of different methods are shown in Table 1. Our method achieves the biggest PSNR and SSIM except the VDCNN(CNN-based method), indicating its superiority in performance.

To investigate the performance of the different parts in proposed method, we test the algorithm with no texture edges, no scalemap and no pixel selected respectively. We use the dataset to set up the experiment. One of the testing results

compared with ground-truth in Table 2 and the natural images in Fig. 15 shows the difference in respective part.

4.2 Applications

Texture filtering method is the basis of many image processing applications and has a good application prospect [4], [43].

Image detail enhancement could make image details clearer and enables us to better distinguish image details. First, we can obtain the detail layer by subtracting the filtered image from the original image. Then, we can get the detail enhanced image by magnifying and adding back the detail layer to the original image. Figure 16 shows examples of detail enhancement produced by RegCov, TF and ours. If the edges of the smoothed image are heavily blurred, there will be halos along the blurred edges in the detail enhanced image, as indicated by the red arrow in Fig. 16b, while, if the edges are sharper than those in the original image, the edges in the detail enhanced image will cause the gradient reversals, as indicated by the yellow arrow in Fig. 16c. Our
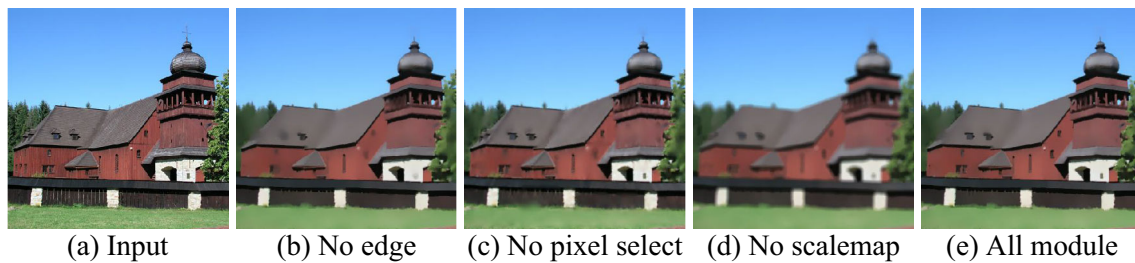
(a) Input   (b) No edge   (c) No pixel select   (d) No scalemap   (e) All module

Fig. 15 Ablation study. **a** Input images. **b** Image smoothing result with no texture edges. **c** Image smoothing result with no pixel selected. **d** Image smoothing result with no scalemap. **e** All module

Fig. 16 Results and comparison on detail enhancement. The parameters set are as follows: RegCov [$k = 4$, $\sigma = 0.2$, $Model1$], and TF [$\sigma = 0.1$, $\sigma_s = 3$]
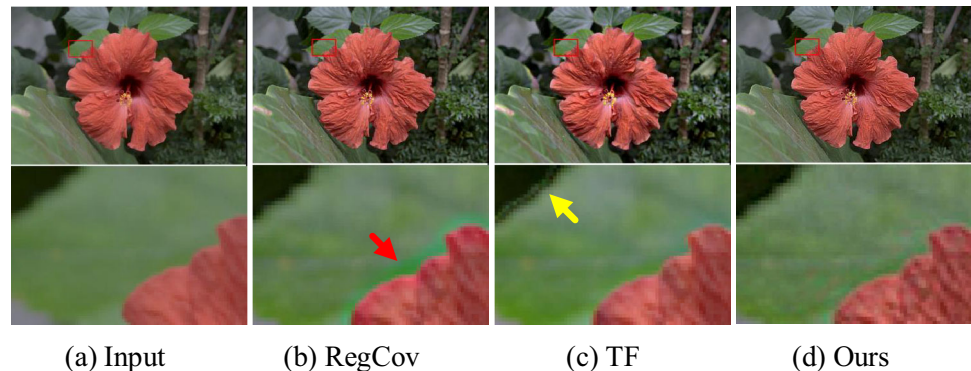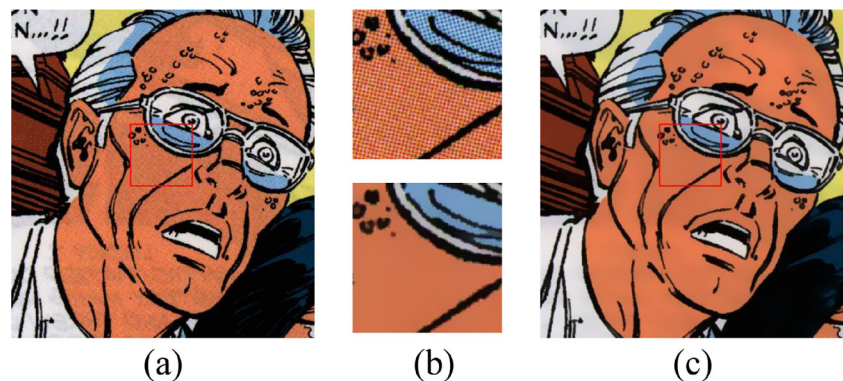

(a) Input   (b) RegCov   (c) TF   (d) Ours

Fig. 17 Inverse halftoning example. **a** Input image. **b** Close-ups. **c** Our filtered result

(a)   (b)   (c)

method can produce the results free of halos and gradient reversals as illustrated in Fig. 16d.

In addition, there are some other applications of our method. Figure 17 shows an application of inverse halftoning, which aims to transform halftone images composed of stipple dots to continuous-tone images. The operation is essentially texture filtering and our method shows a good performance in terms of removing stipple dots while keeping important structures. Figure 18 shows the results of virtual contour restoration. Natural images often contain objects with no clear boundaries, but these boundaries with large contrast can be perceived by human visual systems. Our method can restore these originally nonexistent edges reflecting texture change. Some texture image segmentation results by our filtering method are shown in Fig. 19. Our method can effectively suppress different forms of textures while maintaining the main structure, which is useful in texture image segmentation.

5 Conclusion

We propose a novel texture filtering method based on adaptive kernel scales. Our method relies on texture edges to separate structures from textures and to guide image smoothing with multiscale filter kernels. That is, for per pixel, the average-filtered result on its adaptive-scale circular neighborhood is used to represent the pixel. The texture edges are used to select the per-pixel filter scale that the closer a pixel is to the texture edge, the smaller the filter scale is, and the pixels in textural regions have the largest filter scale. This scheme ensures that the filter scale is big enough to eliminate noise, texture, and clutter, but small enough to preserve all structure

Fig. 18 Virtual contour restoration. (top) Input images. (middle) Our filtered results. (bottom) Virtual contour restoration results
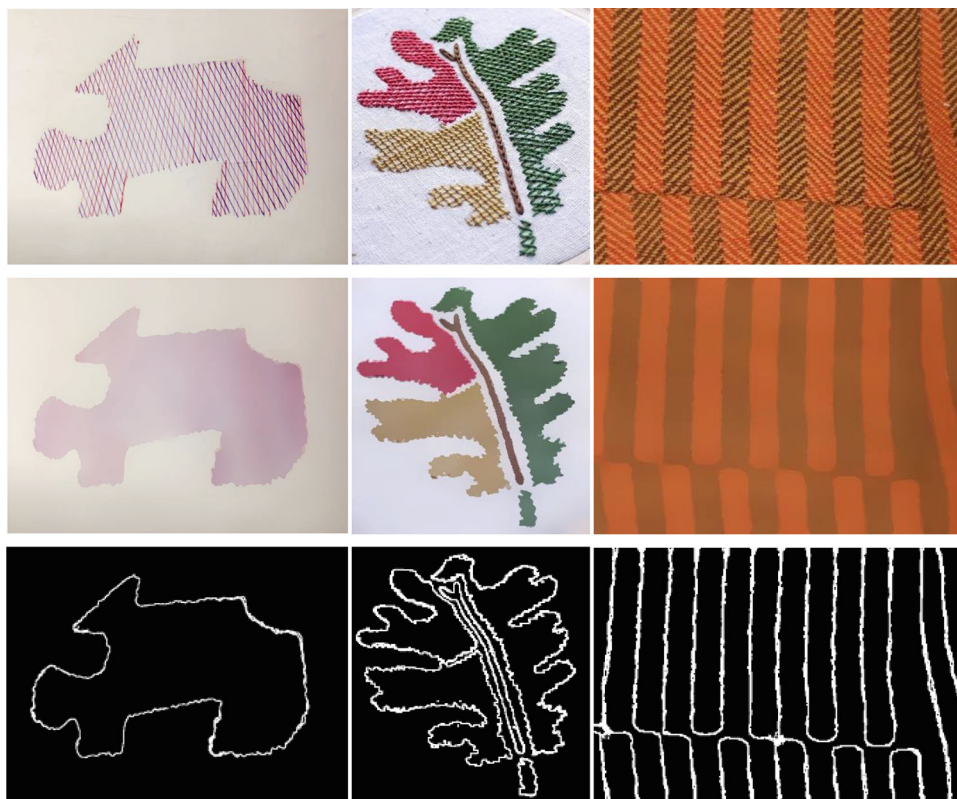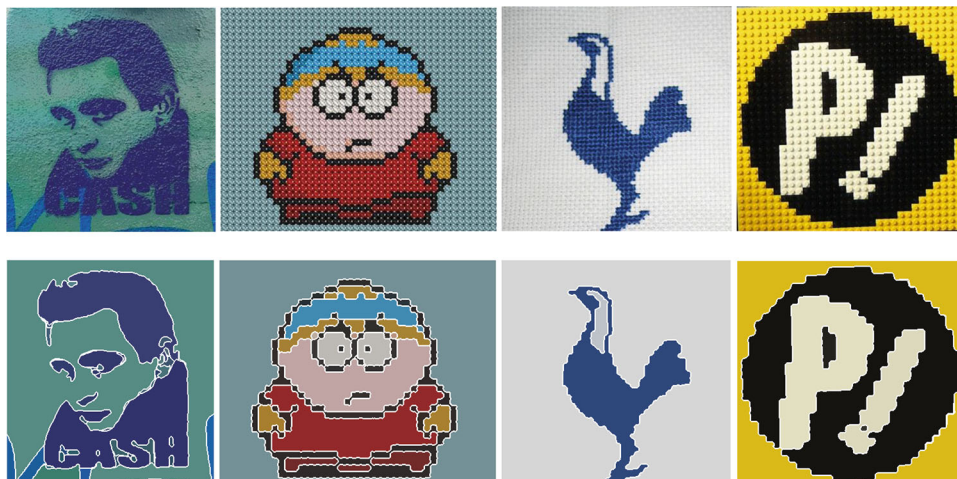

Fig. 19 Texture image segmentation. (top) Input images. (bottom) Segmentation results

edges and corners. Finally, we propose a pixel-selected filter as post-processing to optimize the filtered results. To verify the performance of this proposed method, many comparative tests were conducted with the state-of-the-art methods. The experimental results show that our method has a better performance in filtering out complex textures while preserving the weak main structure.

Declarations

Conflict of interest The authors declare that they have no conflict of interest.

References

1. Kou, F., Wei, Z., Chen, W., Wu, X., Wen, C., Li, Z.: Intelligent detail enhancement for exposure fusion. IEEE Trans. Multimed. **20**(2), 484–495 (2017)

2. Zhou, Z., Wang, B., Ma, J.: Scale-aware edge-preserving image filtering via iterative global optimization. IEEE Trans. Multimed. **20**(6), 1392–1405 (2017)

3. Şener, O., Ugur, K., Alatan, A.A.: Efficient mrf energy propagation for video segmentation via bilateral filters. IEEE Trans. Multimed. **16**(5), 1292–1302 (2014)

4. Xu, L., Yan, Q., Xia, Y., Jia, J.: Structure extraction from texture via relative total variation. ACM Trans. Graph. (TOG) **31**(6), 139 (2012)

5. Cho, H., Lee, H., Kang, H., Lee, S.: Bilateral texture filtering. ACM Trans. Graph. (TOG) **33**(4), 128 (2014)

6. Zhang, C., Ge, L., Chen, Z., Li, M., Liu, W., Chen, H.: Refined tv-l1 optical flow estimation using joint filtering. IEEE Trans. Multimed. **22**(2), 349–364 (2019)

7. Gao, Y., Hu, H.M., Li, B., Guo, Q.: Naturalness preserved nonuniform illumination estimation for image enhancement based on retinex. IEEE Trans. Multimed. **20**(2), 335–344 (2017)

8. Ghosh, S., Gavaskar, R.G., Panda, D., Chaudhury, K.N.: Fast scale-adaptive bilateral texture smoothing. IEEE Trans. Circuits Syst. Video Technol. **30**(7), 2015–2026 (2020)

9. He, K., Sun, J., Tang, X.: Guided image filtering. IEEE Trans. Pattern Anal. Mach. Intell. **35**(6), 1397–1409 (2012)

10. Farbman, Z., Fattal, R., Lischinski, D., Szeliski, R.: Edge-preserving decompositions for multi-scale tone and detail manipulation. ACM Trans. Graph. (TOG) ACM **27**, 67 (2008)

11. Chen, B., Jung, C., Zhang, Z.: Variational fusion of time-of-flight and stereo data for depth estimation using edge-selective joint filtering. IEEE Trans. Multimed. **20**(11), 2882–2890 (2018)

12. Xu, P., Wang, W.: Structure-aware window optimization for texture filtering. IEEE Trans. Image Process. **28**(9), 4354–63 (2019)

13. Deng, G.: Edge-aware bma filters. IEEE Trans. Image Process. **25**(1), 439–454 (2015)

14. Chen, X., Kang, S.B., Jie, Y., Yu, J.: Fast edge-aware denoising by approximated patch geodesic paths. IEEE Trans. Circuits Syst. Video Technol. **25**(6), 897–909 (2015)

15. Eun, H., Kim, C.: Superpixel-guided adaptive image smoothing. IEEE Signal Process. Lett. **23**(12), 1887–1891 (2016)

16. Li, Z., Zheng, J., Zhu, Z., Yao, W., Wu, S.: Weighted guided image filtering. IEEE Trans. Image Process. **24**(1), 120–129 (2014)

17. Lin, T.C.: A new adaptive center weighted median filter for suppressing impulsive noise in images. Inf. Sci. **177**(4), 1073–1087 (2007)

18. Liu, W., Zhang, P., Lei, Y., Huang, X., Yang, J., Reid, ID.: A generalized framework for edge-preserving and structure-preserving image smoothing. In: National Conference on Artificial Intelligence (2020)

19. Liu, W., Zhang, P., Lei, Y., Huang, X., Yang, J., Ng, MKP.: A generalized framework for edge-preserving and structure-preserving image smoothing. IEEE Trans. Pattern Anal. Mach. Intell. pp 1 (2021)

20. Tomasi, C., Manduchi, R.: Bilateral filtering for gray and color images. IEEE Int. Conf. Comput. Vision **98**, 2 (1998)

21. Perona, P., Malik, J.: Scale-space and edge detection using anisotropic diffusion. IEEE Trans. Pattern Anal. Mach. Intell. **12**(7), 629–639 (1990)

22. Liu, W., Zhang, P., Huang, X., Yang, J., Shen, C., Reid, I.: Real-time image smoothing via iterative least squares. ACM Trans. Graph. **39**(3), 1–24 (2020)

23. Liu, W., Chen, X., Shen, C., Liu, Z., Yang, J.: Semi-global weighted least squares in image filtering. In: 2017 IEEE International Conference on Computer Vision (ICCV), IEEE Computer Society, pp 5862–5870 (2017)

24. Liu, W., Zhang, P., Chen, X., Shen, C., Huang, X., Yang, J.: Embedding bilateral filter in least squares for efficient edge-preserving image smoothing. IEEE Trans. Circuits Syst. Video Technol. **30**(1), 23–35 (2020)

25. Karacan, L., Erdem, E., Erdem, A.: Structure-preserving image smoothing via region covariances. ACM Trans. Graph. (TOG) **32**(6), 176 (2013)

26. Zhang, Q., Shen, X., Xu, L., Jia, J.: Rolling guidance filter. In: European conference on computer vision, Springer, pp 815–830 (2014)

27. Lin, T.H., Way, D.L., Shih, Z.C., Tai, W.K., Chang, C.C.: An efficient structure-aware bilateral texture filtering for image smoothing. Comput. Graph. Forum Wiley Online Libr. **35**, 57–66 (2016)

28. Jain, P., Tyagi, V.: An adaptive edge-preserving image denoising technique using tetrolet transforms. Vis. Comput. **31**(5), 657–674 (2015)

29. Bao, L., Song, Y., Yang, Q., Yuan, H., Wang, G.: Tree filtering: efficient structure-preserving smoothing with a minimum spanning tree. IEEE Trans. Image Process. **23**(2), 555–569 (2013)

30. Zhang, F., Dai, L., Xiang, S., Zhang, X.: Segment graph based image filtering: Fast structure-preserving smoothing. In: Proceedings of the IEEE International Conference on Computer Vision, pp 361–369 (2015)

31. Yu, L.H., Feng, Y.Q., Chen, W.F.: Adaptive regularization method based total variational de-noising algorithm. J. Image Graph. **14**(10), 1950–4 (2009)

32. Xu, L., Lu, C., Xu, Y., Jia, J.: Image smoothing via L_0 gradient minimization. ACM Trans. Graph. TOG ACM **30**, 174 (2011)

33. Ham, B., Cho, M., Ponce, J.: Robust image filtering using joint static and dynamic guidance. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp 4823–4831 (2015)

34. Li, Y., Huang, JB., Ahuja, N., Yang, MH.: Deep joint image filtering. In: European Conference on Computer Vision, Springer, pp 154–169 (2016)

35. Kim, Y., Ham, B., Do, M.N., Sohn, K.: Structure-texture image decomposition using deep variational priors. IEEE Trans. Image Process. **28**(6), 2692–2704 (2018)

36. Zhu, F., Liang, Z., Jia, X., Zhang, L., Yu, Y.: A benchmark for edge-preserving image smoothing. IEEE Trans. Image Process. **28**(7), 3556–3570 (2019)

37. Comaniciu, D., Meer, P.: Mean shift: A robust approach toward feature space analysis. IEEE Trans. Pattern Anal. Mach. Intell. **24**(5), 603–619 (2002)

38. Martin, D., Fowlkes, C., Malik, J.: Learning to detect natural image boundaries using local brightness, color, and texture cues. IEEE Trans. Pattern Anal. Mach. Intell. **26**(5), 530–549 (2004)

39. Liu, Y., Liu, G., Liu, C., Sun, C.: A novel color-texture descriptor based on local histograms for image segmentation. IEEE Access **7**(160), 160683–160695 (2019)

40. Liu, Y., Liu, G., Liu, H., Liu, C.: Structure-aware texture filtering based on local histogram operator. IEEE Access **8**, 43838–43849 (2020)

41. Zhang, Z., He, H.: A customized low-rank prior model for structured cartoon-texture image decomposition. Signal Process. Image Commun. **96**(8), 116308 (2021)

42. Wang, Z., Bovik, A., Sheikh, H., Simoncelli, E.: Image quality assessment: from error visibility to structural similarity. IEEE Trans. Image Process. **13**(4), 600–612 (2004)

43. Su, Z., Luo, X., Deng, Z., Liang, Y., Ji, Z.: Edge-preserving texture suppression filter based on joint filtering schemes. IEEE Trans. Multimed. **15**(3), 535–548 (2012)

Guanyu Zhang received the M.S. and Ph.D. degrees in mechanical engineering from Jilin University, in 2015. He is an associate professor in the Department of Instrument Science and Electrical Engineering, Jilin University. His research interests include mechatronics system design and artificial intelligence control technology research

Beichen Sun received the B.E. degree from Kunming University of Science and Technology, China, in 2019, and is working towards a M.E. degree at Jilin University. His research interests include computer vision, image processing, texture filtering and image fusion.

Yang Liu received the Ph.D. degree from Jilin University in 2020 and is working as an assistant research fellow in the Department of Instrument Science and Electrical Engineering, Jilin University. His research interests include stereo vision, 3D reconstruction, texture filtering and SLAM.

Yuehan Qi is a student pursuing a Bachelor of Computing(Hons.) degree at Queen's University, Ontario, Canada. Her current research interests includes machine learning and computer vision.