Computational Visual Media
DOI 10.1007/s41095-xxx-xxxx-x Vol. x, No. x, month year, xx–xx

Research Article

Image smoothing based on global sparsity decomposition and variable parameter

Xiang Ma[1], Xuemei Li[1,2], Yuanfeng Zhou[1] and Caiming Zhang[1,2,3] (✉)

Abstract Smoothing images, especially with rich texture, is an important problem in computer vision. To obtain an ideal result is difficult due to complex, irregular, and anisotropic of the texture. Besides, some common properties are possessed by the texture and the structure in an image. It is hard to compromise in remaining structure and removal of texture simultaneously. To create an ideal algorithm of smoothing image, we face three problems: For images with rich textures, the smoothing effect is expected to be enhanced, improve the inconsistency of the smoothing results in different parts in an image, and it is necessary to create a method of evaluating the smoothing effect. We apply texture pre-removal based on global sparse decomposition with variable smoothing parameter to solve the first two problems. A parameter surface constructed by an improved Bessel method is used to determine the smoothing parameter. Three evaluation rates: edge integrity rate, texture removal rate, and gradient value distribution are proposed to cope with the third problem. We use Alternating Direction Method of Multipliers (ADMM) to complete the whole algorithm and obtain the results. Experiments show that our algorithm is better than the existing algorithm in visual effect and quantitative index. We also demonstrate our method's ability in other applications such as clip-art compression artifact removal and content-aware image manipulat.

1 Shandong University, Jinan, 250101, China. E-mail: sdu_max@163.com (X. Ma), xmli@sdu.edu.cn (X. Li), yfzhou@sdu.edu.cn (Y. Zhou), czhang@sdu.edu.cn (C. Zhang ✉).

2 Shandong Co-Innovation Center of Future Intelligent Computing, Yantai, 264025, China.

3 Digital Media Technology Key Lab of Shandong Province, Jinan, 250014, China.

Manuscript received: 2020-12-30; accepted: -.

Keywords image smoothing, texture removal, global sparse decomposition, Bessel method.

1 Introduction

Nature images usually contain texture and structure. The human visual system can easily understand the structure without being affected by texture. However, for the computer, because the texture can be complex, irregular, and anisotropic [24], it is a challenging task to remove the texture from the image. The purpose of image smoothing is to remove the texture without destroying the structure as much as possible. Image smoothing is an important and widely used image processing technology, such as image segmentation, edge extraction, image enhancement, image decomposition, and artifact removal, to simplify the problem immensely. The existing image smoothing algorithms can be roughly divided into three categories: filter based on local information, global optimization framework, and data-driven method.

Filter based on local information: Bilateral filtering (BLF) [25] is a representative smoothing filter, which achieves smoothing by estimating the value of local patches by weighted average (Gaussian kernel estimation). After BLF was proposed, many improved versions [4, 27] appeared, mostly modified Gaussian kernels. Among them, bilateral texture filtering (BTF) [6] can ensure that the high sharpness of edges, but the flat regions are not regular enough, and the visual effect is poor. Tree filtering [1] successfully mitigates the ringing phenomenon by constructing a tree structure, but if a misclassified pixel occurs, it causes the main edge to break down, resulting in a false boundary. Filters based on local information also include: anisotropic filter [20], guided filter [11], extremum smoothing algorithm [22], etc. Most of these filters are intuitive and simple, but are too dependent on local information and cause ringing phenomenon.

Global optimization framework: Weighted least squares (WLS) [8] is a relatively robust image smoothing algorithm based on global optimization, particularly suitable for image gradual coarsening and edge preserving multi-scale details extraction. Inspired by the feature extraction algorithm Difference-of-Gaussian (DoG) [14, 17], Relativity-of-Gaussian (RoG) [3] smoothes the image by describing the relationship between Gaussian filters with different sizes. However, RoG has a series of problems, such as difficult parameter control and easy local information loss. A well-known algorithm, total variation (TV) [21] performing regular optimization based on the L_1 norm, is often used in image denoising and restoration, but cannot effectively achieve smoothing. By approximately highlighting the image structure to control the number of non-zero gradients, L_0 gradient minimization (L_0) [28] algorithm has adequate protection for the main edge, but easy to lose more original color, lacking aesthetic. L_0 gradient projection (L_0p) [19] solves hard to control the parameters of L_0, without limiting the obvious pseudo-boundaries. To overcome the shortcoming of L_0p, algorithm [18] restricts the smoothed image's gradient, only matching a few images. Compared with the filter method, the optimization framework is more flexible but lacks local information protection, especially easy to lose the local weak edge. The relative total variation (RTV) [30] algorithm applies the relative norm to combine the local filter and global optimization framework. Although the effect is sound, it may cause edge expansion and fail to protect the local weak edge.

Data-driven method: With the development of deep learning, data-driven image smoothing algorithms are presenting gradually. However, as there is no ground truth in image smoothing, conventional supervised and semi-supervised learning methods cannot be well used. The algorithms [5, 29] attempt to use a unified CNN framework to simulate the previous smoothing methods [9, 30], without getting rid of the limitations of the original algorithms. Although the algorithm (DVP) [13] optimizes the image smoothing process to improve the effect to a certain extent by training parameters, generalization is always the barrier.

Most filters based on local information are relatively intuitive and simple. Nonetheless, they tend to depend on the image's local information, resulting in ringing, edge expansion and other phenomena; The optimization framework methods are flexible, but the regular terms' global selection can hardly protect local information; Generalization is still a limitation of the data-driven methods. Combining the local filter and the global optimization framework can mitigate the problem to some extent, there are still incomplete, such as insufficient weak-edge protection.

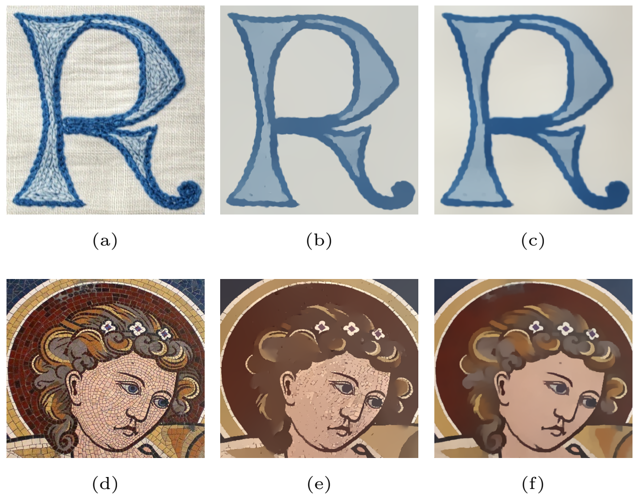


Fig. 1 Examples of image smoothing: (a),(d) original images; (b),(e) L_0 [28] smoothing results; (c),(f) ours results

After research, we summarize three main problems faced by image smoothing at the present stage, and propose improvement schemes:

(1) **The smoothing effect of rich texture image is expected to be enhanced:** In general, with the increase of image texture, the algorithm's smoothing effect becomes worse, as shown in Fig.1. So we set the global sparse regular term to decompose the image into tow parts and remove texture part to improve the performance.

(2) **The inconsistency of the smoothing results in different parts needs to improve:** Due to uneven illumination, contrast, and other factors in the image, the smoothing parameters should change in different parts of it. Therefore, for the local adaptive parameters the patch-shift is used. More importantly, to solve the pseudo-boundary, we propose to construct the parametric fitting surface so that allow continuous parameter variation throughout the region and improve efficiency.

(3) **An evaluating method of the smoothing effect is necessary:** Without ground truth, image smoothness cannot be directly evaluated by PSNR, SSIM, and other conventional indexes. Comparing different algorithms based on visuals alone is too subjective, so there is an urgent need for quantitative

indexes as evaluation criteria. The difficulty with image smoothing lies in the algorithm's ability to distinguish between texture and edge, but the human eye can readily achieve this. Therefore, we can compare the edges extracted from the smoothing results with the manually selected edges from the original image to evaluate the smoothing effect. Besides, image smoothing changes the gradient distribution, and the gradient is positively correlated with image smoothness, so we can compare the gradient distribution of the results to evaluate the effect of algorithms. Based on the above two points, we propose three indexes such as edge integrity rate, texture removal rate and gradient value distribution to evaluate the results from edge and gradient quantitatively.

In summary, we combine local filters with the global optimization framework and propose an image smoothing algorithm based on global sparsity decomposition and variable parameter. Firstly, the global sparse decomposition is used to pre-remove part of the texture to improve the smoothing performance. The variable parameter is then obtained as the parametric surface by patch-shift selection with the improved Bessel method to ensure localization and continuity. Finally, to limit the image gradient variation through the L_1 norm, we achieve image smoothing. The flow chart is shown in Fig.2.

The main structure of this paper is as follows: Section II introduces the model framework of our algorithm, as well as the global sparse decomposition, patch-shift parameter selection and improved Bessel fitting; Section III describes in detail the solution of our algorithm based on Alternating Direction Method of Multipliers (ADMM); Section IV shows the effect of selecting different parameters and compares the differences in visual effects and quantitative index between other algorithms and ours. Section V introduces the application of our algorithm in clip-art compression artifact removal and content-aware image manipulation. Finally, Section VI summarizes the paper.

2 Problem Formulation

This section introduces how to solve the first two problems. Nature image can generally be described as:

$$y = x + n$$

Here y , x, and n represent the original image, structure, and texture respectively. The goal of image smoothing is to obtain x from y. The global

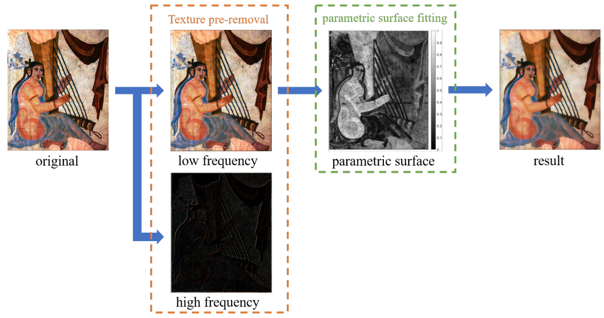


Fig. 2 Flowchart of proposed algorithm

optimization framework can be described as:

$$\hat{x} = arg\,\underset{x}{min}\,\frac{1}{2}\|y - x\|_2^2 + \lambda R(x) \tag{1}$$

$\hat{x}$ is the result, the first term in this formula is the fidelity term, and λ is the smoothing parameter. $R(x)$ is the regular term, which is prior information and non-negative. It is worth noting that there is a need to satisfy $\lambda > 0$, otherwise $R(x)$ may not give the right guidance. For example, when $\lambda = -1$, the latter term in Eq.1 is $\| - \bigtriangledown x\|_2^2$ allows us expect a larger gradient of x when solving the minimum, which runs counter to our intention of removing texture information.

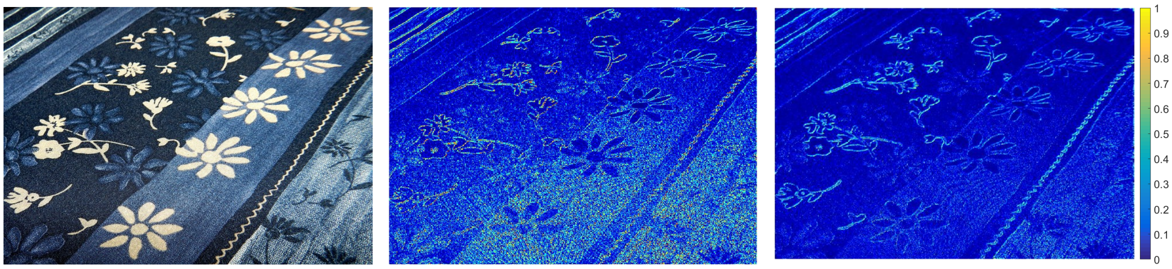
2.1 Global Sparse Decomposition

For the first problem, we decompose the image into two parts: low frequency representing the structure and the high frequency containing the texture. In image super-resolution and image reconstruction, high frequency is usually considered as the missing information in the scaling process to refine the result [15, 31, 34].

In contrast, high frequency needs to be removed during image smoothing. What needs to be made clear here is that we need to remove the texture beforehand and ensure that the edges are not damaged as much as possible. Therefore, global sparse decomposition has been chosen to assure that the high frequency is sparse and to reduce the loss of structural information. The algorithm can be described as:

$$R_{str}(y) = \|y_H\|_1 + \frac{\kappa}{2}\sum_{d=1}^{4}\|\bigtriangledown_d \otimes y_L\|_2^2$$
$$s.t. \;\; y = f_L \otimes y_L + y_H$$

y_L and y_H represent low and high frequency respectively. f_L is a low-pass filter, and $\otimes$ is the convolution operator. $f_L \otimes y_L$ is used to ensure the smooth component contains low-frequency information, so as to ensure y_H approximately represents the texture. κ controls the smoothness level. The larger value of κ, the more information y_H contains. $\bigtriangledown_d$



(a) original image (b) $|\bigtriangledown y|$ (c) $|y_H|$ $(\kappa = 1)$

Fig. 3 Comparison between image gradient and the proposed residual component y_H for $\kappa = 1$ in absolute value

means calculating the gradient in d direction, and $d \in \{1 = horizontal, 2 = vertical, 3 = 45\,degrees, 4 = 135\,degrees\}$. It is well known that L_p norm can promote sparsity when $p \leq 1$. Here we use $\|y_H\|_1$ to force y_H to be the sparse component under L_1 norm (L_1 norm is used to ensure the convexity), making y_H contain only texture without destroying the structure.

We compare high frequency y_H with the gradient, as shown in Fig.3, and label different colors according to the pixel values. Obviously, y_H is more sparse than the gradient. We further analyze this property in Fig.4, and present the numerical distribution of gradient and y_H with different κ. It can be seen that the peak value of y_H is near 0, and the numerical distribution is closer to the Laplace distribution. This is since y_H is treated sparsely under the L_1 norm. Comparatively, the gradient's numerical distribution is fluctuant, and the peak is non-zero, which contains much missing structural information from image. Furthermore, it can be presumed that κ can affect the sparsity of y_H. The larger κ is, the more sparse y_H is. After removing y_H, we use y_L for smoothing.

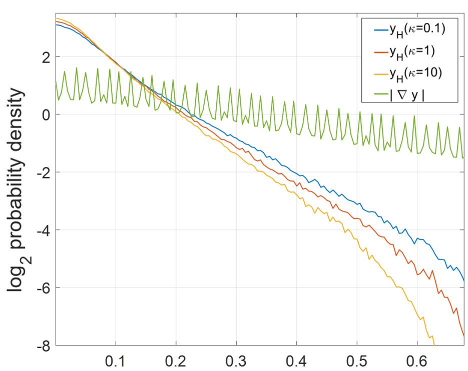


Fig. 4 Image gradient distribution and y_H distribution with different κ values

2.2 Patch-shift Parameter Selection and Parametric Surface Fitting Based on Improved Bessel

The second problem is mainly because λ is a constant parameter. Even if we get the global optimal solution, it may not satisfy the local optimal. Separate parameter calculations for all points can validly solve this problem but are incredibly time-consuming, so we propose a two-step parameter calculation algorithm, including patch-shift parameter selection and parametric surface fitting.

2.2.1 Patch-shift parameter selection

Patch-shift is an intuitive way, where we set the values of patches by comparing them with global variations. To simply adjust the smoothness, λ_G replacing the original λ is set as an adjustable parameter for users. We define the local parameter as $\lambda_{i,j}$.

$$\lambda_{i,j} = \chi_{i,j}\lambda_G, \quad \chi_{i,j} = se^{-\sigma(y_L)/(\varepsilon+\sigma(\Omega_{i,j}))} \tag{2}$$

$\Omega_{i,j}$ refers to the patch and (i, j) is the coordinate of the patch. $\sigma(\cdot)$ is the standard deviation operation. $\chi_{i,j}$ is the fluctuation rate. s is a simply adjustment factor. ε is a small value to prevent the denominator from being zero. As shown in Fig.5, the smoother the $\Omega_{i,j}$, the smaller and more rapidly decreasing $\chi_{i,j}$. Conversely, $\chi_{i,j}$ is larger and slowly increasing. However, due to the patch-shift parameter selection discontinuity, the results show an obvious pseudo boundary at the junction of patches, as shown in Fig.6(d).

2.2.2 Parametric surface fitting

To solve this problem, we propose a novel algorithm: Assign the patches' parameters to their center point to get a set of sample values. This set of values corresponding horizontally and vertically can be considered a low-resolution surface, and we fit the parametric surface based on it, where each pixel can get a unique parameter. After comparing various fitting methods, the Bessel method was finally chosen. There

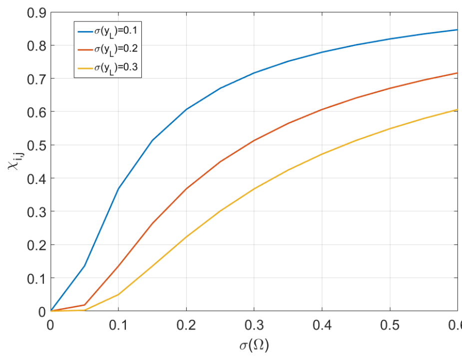
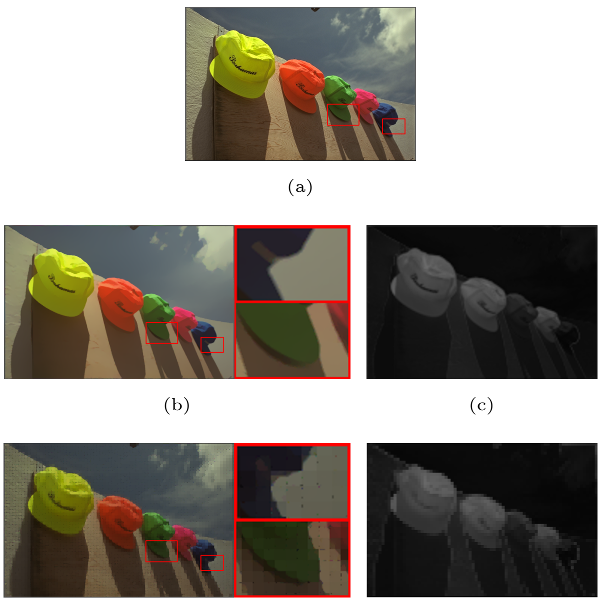


Fig. 5 Image gradient distribution and y_H distribution with different κ values

are two reasons: (1) The sample values calculated by Eq.2 tend to fluctuate and cannot well-tuned by s alone. Bessel typically smoothes the midpoint by passing only the starting and ending points of the sample values, which allows for easy parameter adjustment; (2) Since the Bessel method guarantees convexity of the curve, parameter λ can satisfy $\lambda > 0$ to ensure correctness.



(a)

(b) (c)

(d) (e)

Fig. 6 Comparisons of parametric surface with or without Bessel: (a) original image; (b) smoothed result with improved Bessel; (c) parametric surface with Bessel; (d) smoothed result without improved Bessel; (e) parametric surface without Bessel

Usually, the highest similarity is found between adjacent points, so we propose a fitting method based on neighboring patches (n-patches). To allow more sample values to act on point p which needs to be solved. While considering the computational complexity, the 16 sample values closest to p are chosen

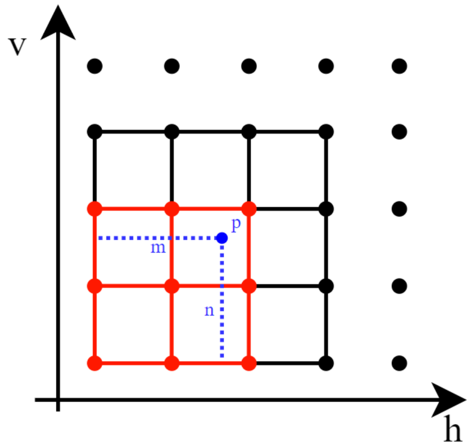


Fig. 7 Bessel method

to construct parametric surface. Each 3×3 patch is called an n-patch, and 16 sample values constitute 4 n-patches, as shown in Fig.7 (All red points construct one n-patch). We assume the window sliding step as 1 for ease of illustration. Here we set $F_{i,j}(p)$ refers to the parametric surface of point p. (i, j) is the coordinate of the nearest sample value at lower left of p and (p_i, p_j) is the coordinate of p. Moreover, we set $f_{i,j}(p)$ as the surface for each n-patch, and use the following function to fit the nine sample values.

$$f_{i,j}(p) = \sum_{h=-1}^{1} \sum_{v=-1}^{1} \varphi_h(m)\varphi_v(n)\chi_{i+h,j+v}, \ 0 \le m, n \le 1$$

$\chi_{i,j}$ refers to the sample values. $m = (p_i - i + 1)/2$, $n = (p_j - j + 1)/2$. Here $\varphi_c(t)$, $c \in \{-1, 0, 1\}$ are Bezier basis functions defined by:

$$\varphi_{-1}(t) = (1-t)^2, \ \varphi_0(t) = 2t(1-t), \ \varphi_1(t) = t^2$$

t is the distance from p to the reference point of $f_{i,j}(p)$ in the space (m, n). All of $f_{i,j}(p)$,$f_{i+1,j}(p)$,$f_{i,j+1}(p)$ and $f_{i+1,j+1}(p)$ can compute different parameters. However, we hope points with the same pixel values to have the same parameters in order to make the image more smoother except the edges. So a pixel-sensitive Gaussian weight considering pixel values is presented to sum up the four parameters. $F_{i,j}(p)$ can be defined as:

$$F_{i,j}(p) = \frac{\sum_{h=0}^{1}\sum_{v=0}^{1} \omega_{i+h,j+v}(p) f_{i+h,j+v}(p)}{\sum_{h=0}^{1}\sum_{v=0}^{1} \omega_{i+h,j+v}(p)} \tag{3}$$

The weight function in Eq.3 is:

$$\omega_{i,j}(p) = \beta_{i,j}(p)(1-m)(1-n)$$
$$\omega_{i+1,j}(p) = \beta_{i+1,j}(p)m(1-n)$$
$$\omega_{i,j+1}(p) = \beta_{i,j+1}(p)(1-m)n$$
$$\omega_{i+1,j+1}(p) = \beta_{i+1,j+1}(p)mn$$

$\beta_{i,j}(p)$ is defined as:

$$\beta_{i,j}(p) = e^{-(P-P_{i,j})^2/(s\delta)},\ \delta = \sum_{i=0}^{1}\sum_{j=0}^{1}(P - P_{i,j})^2$$

P and $P_{i,j}$ are the pixel value of point p and the center point of $f_{i,j}(p)$, respectively. s is a adjustment factor. As shown in Fig.8, ω makes the result more smoother. After obtaining the parameters of all points,

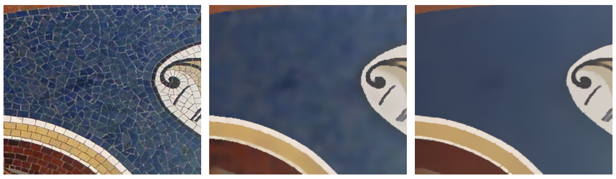


Fig. 8 Comparison of the results with or without ω:(a) original image;(b) smoothing result without ω;(c) smoothing result with ω

we combine them as χ_{y_L}. Under the control of λ_G, the final parameter can be expressed as:

$$\lambda_{y_L} = \chi_{y_L}\lambda_G \tag{4}$$

3 Efficient ADMM Method for Image Smoothing

In order to improve the efficiency, we combine the global sparse decomposition, parametric surface, and L_1 norm to get our model [10, 26].

$$\arg\min_{x,y_L,y_H} \frac{1}{2}\|y-x\|_2^2 + \lambda_{y_L}\sum_d \|\bigtriangledown_d x\|_1 + \alpha\|y_H\|_1 + \frac{\kappa}{2}\sum_d \|\bigtriangledown_d \otimes y_L\|_2^2 \quad s.t.\ \ y = f_L \otimes y_L + y_H \tag{5}$$

Here α and κ weigh the sparsity of y_H. λ_{y_L} controls the sparsity of gradient. Eq.5 is non-differentiable and non-linear and is difficult to solve directly. So, we adapt ADMM to optimize this function iteratively. Two Lagrange constraints are added based on this strategy:

$$\arg\min_{x,y_L,y_H,T} \frac{1}{2}\|y-x\|_2^2 + \lambda_{y_L}\|T\|_1 + \alpha\|y_H\|_1 + \frac{\kappa}{2}\|\bigtriangledown \otimes y_L\|_2^2 + \frac{\gamma_1}{2}\|y-(f_L\otimes y_L + y_H) - \mu_1\|_2^2 + \frac{\gamma_2}{2}\|T - \bigtriangledown x - \mu_2\|_2^2 \tag{6}$$

For ease of writing, we omit d and replace the four-directions operator with $\bigtriangledown$. γ_1 and γ_2 are the parameters of the two Lagrange constraints. In practical, γ_1 and γ_2 are initialized to small positive values and are increased in each iteration to ensure convergence. μ_1 and μ_2 are Lagrange multipliers. T is the auxiliary parameter. Eq.6 is convex, so we can update each parameter iteratively until convergence.

3.1 Solver

3.1.1 Computing y_L

Assuming all other parameters are fixed, we can get:

$$\arg\min_{y_L} \frac{\kappa}{2}\|\bigtriangledown \otimes y_L\|_2^2 + \frac{\gamma_1}{2}\|y - (f_L\otimes y_L + y_H) - \mu_1\|_2^2$$

The above formula can be solved directly by gradient descent and optimized by two-dimensional fast Fourier transform:

$$y_L = \mathcal{F}^{-1}\left(\frac{\gamma\cdot\overline{\mathcal{F}(f_L)}\mathcal{F}(y-y_H-\mu_1)}{\kappa\cdot\overline{\mathcal{F}(\bigtriangledown)}\mathcal{F}(\bigtriangledown) + \gamma_1\cdot\overline{\mathcal{F}(f_L)}\mathcal{F}(f_L)}\right) \tag{7}$$

$\mathcal{F}(\cdot)$ and $\mathcal{F}^{-1}(\cdot)$ represent Fast Fourier Transform (FFT) and Inverse Fast Fourier Transform (IFFT). $\overline{\mathcal{F}(\cdot)}$ is complex conjugation operator. We invert the matrix in the space domain into element multiplication in the frequency domain, making the operation more efficient.

3.1.2 Computing y_H

Consistent with the idea of Subproblem1 and let $\Lambda = y - f_L\otimes y_L + \mu_1$, we can get the formula of y_H:

$$\arg\min_{y_H} \alpha\|y_H\|_1 + \frac{\gamma_1}{2}\|y_H - \Lambda\|_2^2$$

This problem can be solved independently for each pixel i via a simple soft-thresholding:

$$[y_H]_i = sign([y_H]_i)\cdot max(0, [\Lambda]_i - \frac{\alpha}{\gamma_1}) \tag{8}$$

3.1.3 Computing T

Similarly, the variables other than T in Eq.6 are fixed. The solution of T can be expressed as follows:

$$\arg\min_{T} \lambda_{y_L}\|T\|_1 + \frac{\gamma_2}{2}\|T - \bigtriangledown x - \mu_2\|_2^2$$

By using the same way as Eq.8, it can be obtained that:

$$[T]_i = sign([T]_i)\cdot max(0, [\bigtriangledown x + \mu_2]_i - [\frac{\lambda_{y_L}}{\gamma_2}]_i) \tag{9}$$

3.1.4 Computing x

After solving T, y_L and y_H, the optimization of x can be described as:

$$\arg\min_{x} \frac{1}{2}\|y-x\|_2^2 + \frac{\gamma_2}{2}\|T - \bigtriangledown x - \mu_2\|_2^2$$

The above function also meets the requirements of gradient descent, and can be solved as:

$$x = \mathcal{F}^{-1}\left(\frac{\mathcal{F}(y) + \gamma_2\cdot\overline{\mathcal{F}(\bigtriangledown)}\mathcal{F}(T-\mu_2)}{1+\gamma_2\cdot\overline{F(\bigtriangledown)}\mathcal{F}(\bigtriangledown)}\right) \tag{10}$$

3.1.5 Update μ_1 and μ_2

At the end of each iteration, the Lagrange multipliers need updated:

$$\begin{aligned}\mu_1 &= \mu_1 + (f_L\otimes y_L + y_H - y)\\ \mu_2 &= \mu_2 + (\bigtriangledown x - T)\end{aligned} \tag{11}$$

Algorithm 1 Image Smoothing Based on Global Sparsity Decomposition and Variable Parameter

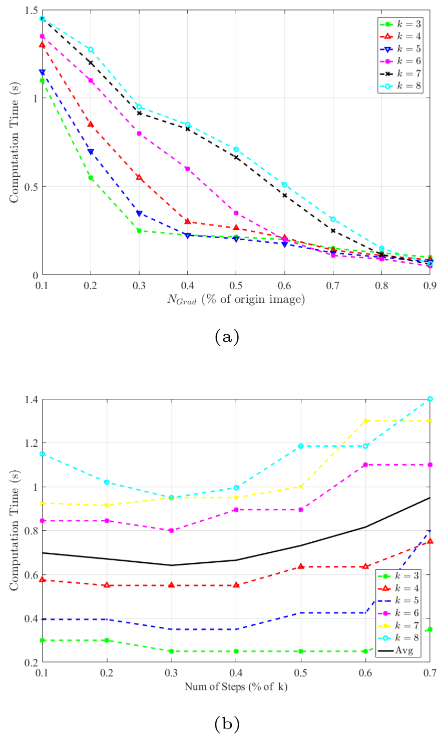
Input:
Original image: y
ADMM parameters: μ_1, μ_2, γ_1, γ_2
Output:
Smoothed image: x
Initialization: $x = y$, $\mu_1 = 0$, $\mu_2 = 0$
while not converged **do**
  Solve Subproblem y_L by computing Eq.7;
  Solve Subproblem y_H by computing Eq.8;
  Obtain adaptive parameter λ_{y_L} from y_L by computing Eq.4;
  Solve Subproblem T by computing Eq.9;
  Updata x by solving Eq.10;
  Updata μ_1 and μ_2 using Eq.11;
end while

3.2 Algorithm Summary and Complexity

The entire reconstruction process is outlined in Algorithm 1. In terms of time complexity, the most time-consuming part is the solution of λ_{y_L}, depending on the number of pixels N, the number of patches K and the patch size k. In general, we do not reduce N during the smoothing beacuse of loss of details. However, K and k can directly affect the fitting effect, so we conduct a series of experiments to balance time and performance. We propose N_{Grad} to describe the smoothing effect.

$$N_{Grad}(x) = \sum_{n=1}^{N} C(|\bigtriangledown x_n|),\ C(i) = \begin{cases} 0, & i = 0 \\ 1, & i > 0 \end{cases}$$

Assuming that image x_1 and x_2 are equally large images, $N_{Grad}(x_1) = N_{Grad}(x_2)$ implies that the two images have the same smoothness. Our algorithm can reduce the gradient at each iteration, so the performance of different K or k can be evaluated by by comparing how much time it takes to smooth the same image to the same N_{Grad}. The experimental images of this part are all taken from BSD500. As can be seen from Fig.9(a) that the time consumption is almost the same with the increase of k when N_{Grad} is large enough. However, as N_{Grad} decreases, the time spent is gradually positively correlated with k. Therefore, k can be adjusted according to specific needs. In this paper, $k = 3$. As image patches are selected in various ways, comparing K is confusing. Assuming that k is fixed, we replace K with the image patch move *steps*, which is interpreted as different percentages k. As shown in Fig.9(b), the operation time decreases first and then increases as *steps* increases. The optimal value is about 0.3. Moreover, a decreasing difference between the λ_{y_L} of two adjacent iterations is witnessed during the experiment. So we set a strategy to reduce the number of λ_{y_L} calculations: After the 10th iteration, we calculate λ_{y_L} every five iterations. Experiments show that this strategy can not only ensure the correctness of our algorithm, but also effectively reduce the calculation time. In terms of convergence, Eq.6 is convex. When the values of γ_1 and γ_2 are large enough, ADMM can ensure that the variables converge [2, 7, 23, 33].



(a)

(b)

Fig. 9 Impact of image patch: (a) patch size k; (b) patch steps;

4 Experiments and Discussion

In this chapter, the value of parameters in our algorithm is firstly discussed. Then we compare other algorithms with ours in terms of visual effect, and we create some images to evaluate the results quantitatively. Finally, the solution to the third problem is given.

4.1 Analysis of parameters

The size of f_L can directly affect the time spent on decomposition, so we conduct statistical experiments based on BSD500 to select the most efficient value. As shown in Fig.10, the time consumption is relatively stable and has the lowest average when the size of f_L is 6×6. α affects the smoothness of y_L and the information contained in y_H. As can be seen from Fig.11, image decomposition can effectively separate the high frequency y_L becomes smoother as α increases. Here we set $\alpha = 5$. Fig.12 exposes a set of smoothed results for different λ_G, and it can be seen that the larger λ_G, the smoother the result. We set $\gamma_1 = \gamma_2$ and each iteration increases by 5% [32].

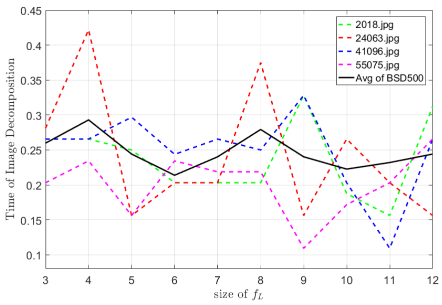
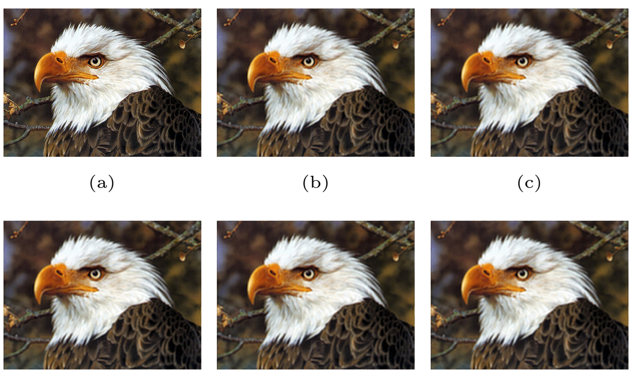


Fig. 10 Impact of f_L in texture pre-removal



(a) (b) (c)

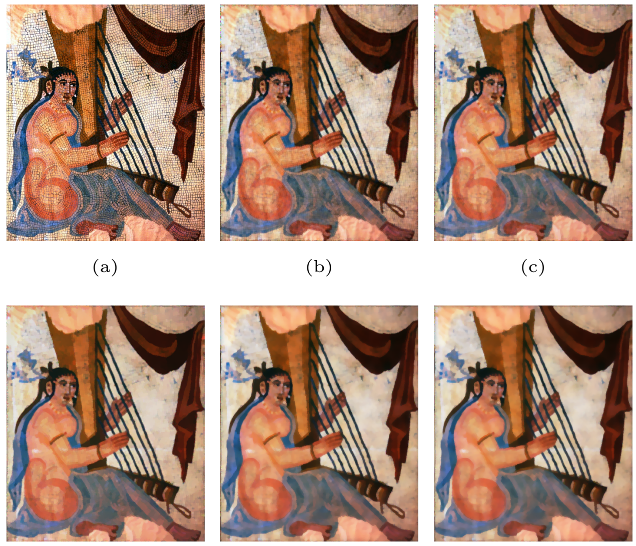
(d) (e) (f)

Fig. 11 decomposition results y_L of difference α: (a) original image; (2) $\alpha = 20$; (3) $\alpha = 40$; (4)$\alpha = 60$; (5)$\alpha = 80$; (6)$\alpha = 100$

4.2 Comparison of Visual Effects

In order to prove the effectiveness of our algorithm, we choose WLS [8], TV [21], Tree filter [1], RoG [3], L_0 [28], RTV [30] and DSHFG [16]. All the algorithms are based on the code provided by authors and manually adjust the parameters.

As shown in Fig.13, WLS doesn't distinguish texture and edge well, nor does TV, and the whole image is very blurry. Tree Filtering averages bilateral weights and Tree weights, but it doesn't protect all edges well. RoG uses several sets of Gaussian kernels with different weights to achieve texture removal, which can fully smooth the image globally, but some edges cannot be well protected. L_0 can better sharpen and protect the strong edges, but the effect of processing high-



(a) (b) (c)

(d) (e) (f)

Fig. 12 Smoothing results of different λ_G: (a) original image; (b) $\lambda_G = 0.001$; (c) $\lambda_G = 0.005$; (d) $\lambda_G = 0.01$; (e) $\lambda_G = 0.02$; (f) $\lambda_G = 0.025$

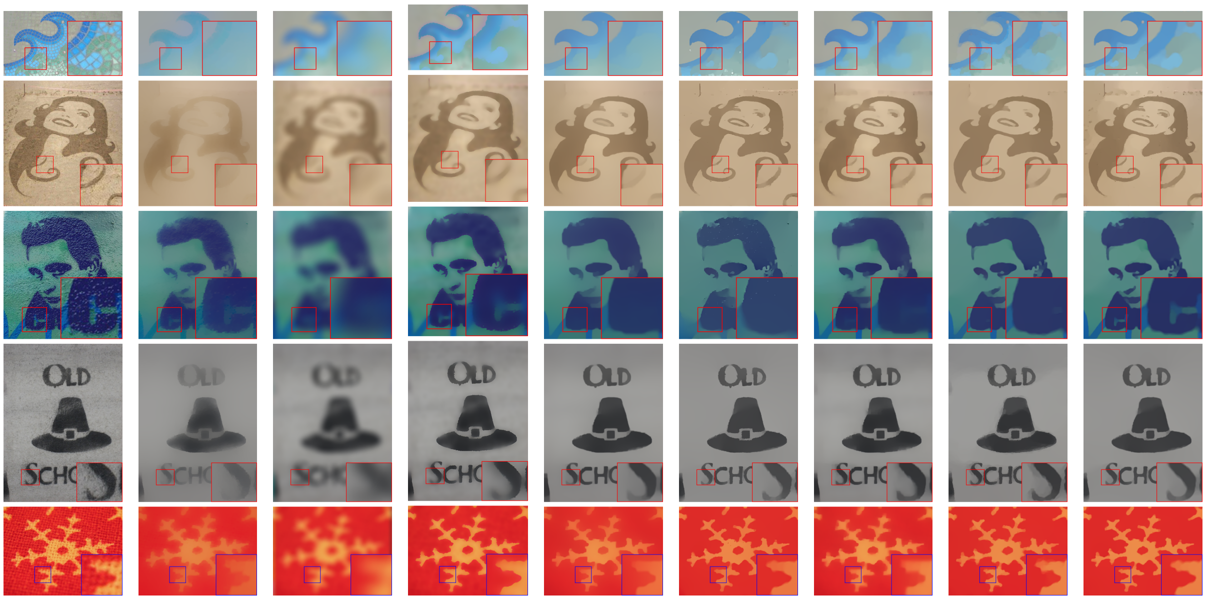
contrast texture images is poor, because it is difficult to distinguish such textures based solely on gradients. RTV's regular term based on local can help it to achieve texture removal, but it cannot protect local weak edge well. DSHFG is an L_0 norm minimization smoothing algorithm based on image decomposition, which removes texture well, but it also loses local weak edge. In contrast, our algorithm can not only distinguish texture and structure well and remove texture, but also effectively protect weak edge.

4.3 Quantitative comparison based on created images

To quantitatively evaluate the results of different algorithms using PSNR, we manually constructed several texture images, as shown in Fig.14. In order to show the poor generalization of data-driven methods, VDCNN[35] and ResNet[35] are added to the control group. The smoothing results are shown in Fig.15 and Table.1. It can be seen that most algorithms except TV can do texture removal well, and there are also some artificial textures have not been removed in Fig.15(d) and Fig.15(e). The PNSR also shows that our algorithm is better.

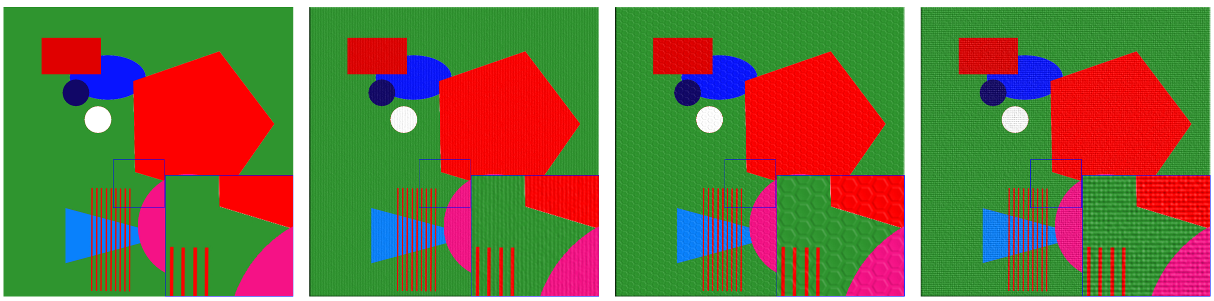
4.4 Quantitative comparison with the proposed evaluation method

We propose three evaluation indexes in terms of edge and gradient distribution to break away the dependence on ground truth.



(a) (b) (c) (d) (e) (f) (g) (h) (i)

Fig. 13 Comparison of visual effects: (a) original image; (b) WLS [8] ($\lambda = 2, \alpha = 2$); (c) TV [21] ($\lambda = 0.08$); (d) Tree [1] ($\sigma = 0.015$); (e) ROG [3] ($\lambda = 0.015, \sigma_1 = 1, \sigma_2 = 3$); (f) L_0 [28] ($\lambda = 0.035, \kappa = 2$); (g) RTV [30] ($\lambda = 0.02, \sigma = 3$); (h) DSHFG [16] ($\lambda = 0.02$); (i) Ours ($\lambda_G = 0.02$)



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

Fig. 14 The created images: (a) is simple created images, the others are the images with different artificial textures.

Tab. 1 Quantitative comparison based on the created image (PSNR)

	TV [21]	RTV [30]	DSHFG [16]	VDCNN [35]	ResNet [35]	Ours
Fig.14(b)	21.2890	30.7315	27.2117	29.7807	29.6161	**31.4621**
Fig.14(c)	21.4065	31.0806	27.6249	30.1331	29.9596	**31.8956**
Fig.14(d)	20.6971	27.9442	25.9466	26.9227	26.6810	**28.2501**
Avg.	21.1309	29.9188	26.9277	28.9455	28.7522	**30.5359**

4.4.1 Edge integrity rate and texture removal rate

Conventional edge extraction algorithm cannot reasonably distinguish texture and edge, as shown in Fig.16(b). So we manually draw the real edges and present the edge integrity rate and the texture removal rate to evaluate the smoothing effect. Edge integrity rate can evaluate the protection of edge. Texture removal rate can evaluate the level of texture removal. Their formula is as follows:

$$EI = \frac{EE(x) \odot GT(y)}{GT(y)}, \; TR = \frac{EE(x) \oplus GT(y)}{EE(y)} \quad (12)$$

Here EI and TR represent edge integrity rate and texture removal rate. $EE(x)$ and $GT(y)$ are the extracted edges from smoothing results and hand-drawn ground truth. The operators $\odot$ and $\oplus$ mean XNOR and XOR, respectively. In fact, the author of

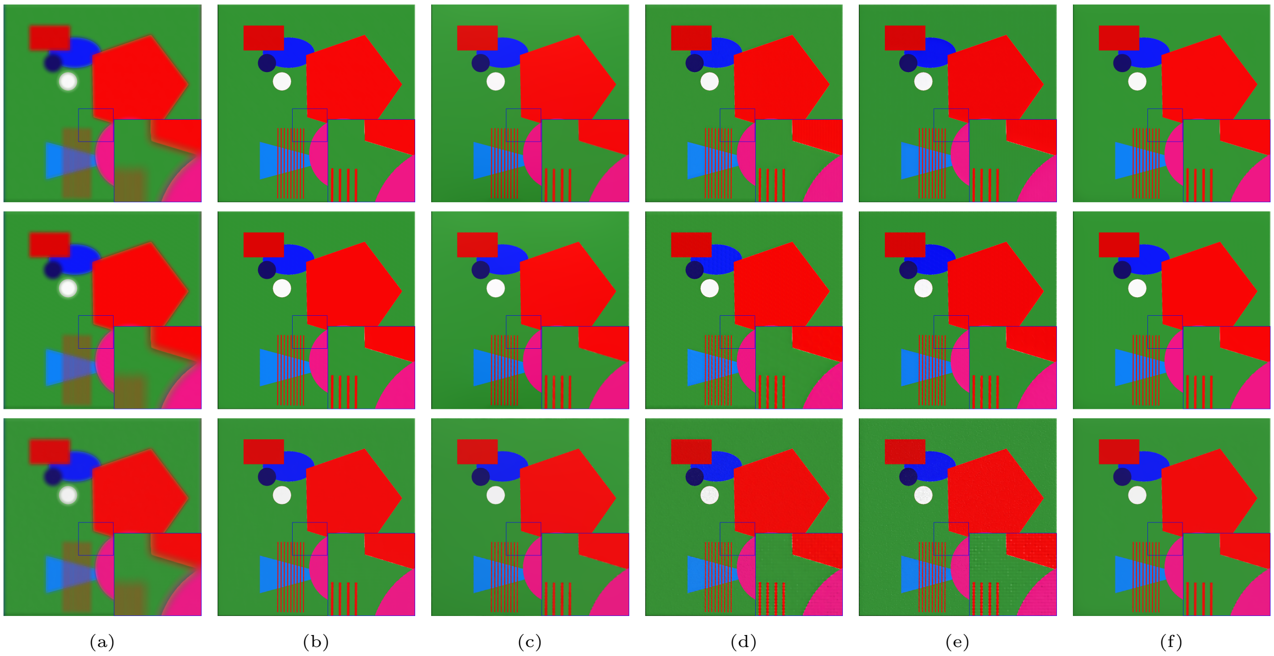


Fig. 15 Comparison of the created images: (a) TV [21] ($\lambda = 0.1$); (b) RTV [30] ($\lambda = 0.015, \sigma = 3$); (c) DSHFG [16] ($\lambda = 0.01$); (d) ResNet [35]; (e)VDCNN [35]; (f)Ours ($\lambda_G = 0.02$). Rows 1 to 3 correspond to Fig.14(b)-Fig.14(d).

RTV provides the texture image we experimented with, but the manual edges are too rough, and we redrew them.

Let us first observe the difference between the several algorithms from the visual effect in Fig.17. As can be seen visually, smoothing can work to simplify edges. WLS and RTV do a good job of removing textures, but they also cause some missing edges. DSHFG can preserve relatively intact edges. However, DSHFG loses some weak edges, such as the flower-like edge at the bottom left of image.

Table.2 and Table.3 show edge integrity rate and texture removal rate of all algorithms and demonstrate that our algorithm outperforms the others. The average edge Integrity rate of all algorithms is lower than 50%. This is because while human eye can determine texture and edge, it can not easy to distinguish the exact location of pixel-level edge. The boundaries obtained by smoothing algorithm are typically 1-3 pixels wide, while the labeled data are generally larger than 3 pixels, which is the problem we will address in our next study. As shown in Table.3, the texture removal rate of RTV, DSHFG and ours are relatively good, and some can even more than 99%. The effects evaluated by the two indexes are consistent with our visual conclusions on the whole, indicating that these two indexes can perform a good quantitative comparison of image smoothing.

Tab. 2 Comparison of edge integrity rate

	TV [21]	RTV [30]	DSHFG [16]	Ours
01_06.jpg	0.0730	**0.6116**	0.5758	0.6018
01_15.jpg	0.0860	0.4422	0.3854	**0.5034**
01_22.jpg	0.1685	0.3945	0.3650	**0.4205**
02_01.jpg	0.1192	0.4094	0.3760	**0.4807**
04_08.jpg	0.1400	**0.5093**	0.4957	0.5050
07_15.jpg	0.0690	0.4153	0.4593	**0.5315**
07_30.jpg	0.1666	0.5575	0.5197	**0.5915**
07_34.jpg	0.2391	0.4631	**0.5472**	0.4951
12_15.jpg	0.1675	0.3350	0.3309	**0.3510**
12_53.jpg	0.2175	0.2800	0.2772	**0.3254**
Avg.	0.1997	0.4385	0.4174	**0.4905**

4.4.2 Gradient value distribution

Image smoothing is about eliminating as much redundant texture as possible, which leads to gradients in the sparse direction. Thus, gradient value distribution can also be used to describe smoothness. On the premise of ensuring that structure is not destroyed, the more the distribution tends to 0, more sparse the gradient is and the smooth effect is better. As shown in Fig.18, the peak values of the gradient for all algorithms are around 0, indicating that the gradients of smoothed images tend to be sparse. Except TV, our algorithm has a higher sparsity. From the visual effect, it can be seen that TV destroys the

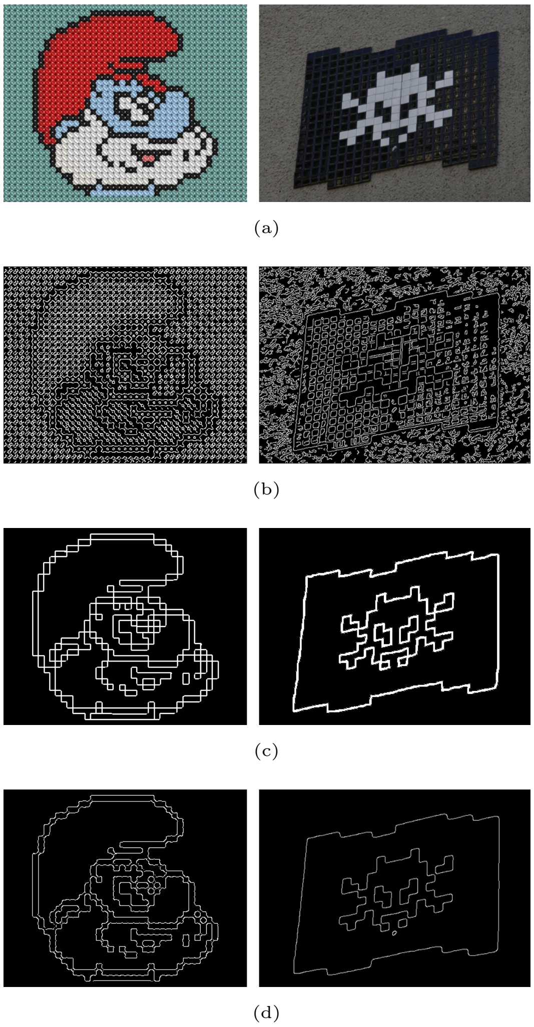


Fig. 16 Comparison of edges before and after smoothing: (a) original images; (b) original edges; (c) ground truth; (d) edges after smoothing

Tab. 3 Comparison of texture removal rate

	TV [21]	RTV [30]	DSHFG [16]	Ours
01_03.jpg	0.1897	0.8450	0.7684	**0.9012**
01_07.jpg	0.3450	**0.9979**	0.9466	0.9968
01_09.jpg	0.3927	0.9339	0.9558	**0.9950**
01_25.jpg	0.4533	0.9417	0.9268	**0.9608**
02_16.jpg	0.3338	0.7933	0.7982	**0.8348**
11_12.jpg	0.4882	0.8957	0.8052	**0.9350**
12_26.jpg	0.4480	0.9511	0.9040	**0.9745**
13_02.jpg	0.4465	0.8712	0.8711	**0.9247**
13_05.jpg	0.6277	0.9879	0.9500	**0.9907**
13_17.jpg	0.4800	0.9868	0.9657	**0.9932**
Avg.	0.4126	0.8622	0.8287	**0.8978**

structural information, which leads to extreme sparsity.

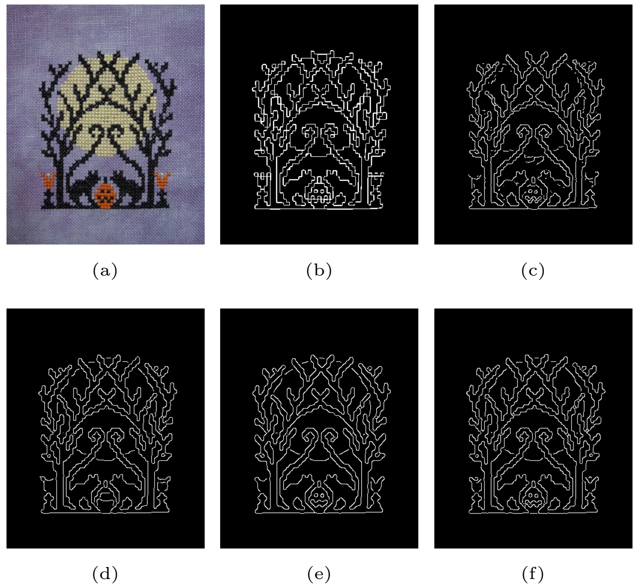


Fig. 17 Comparison of edge extraction: (a) original image; (b) ground truth; (c) WLS [8] ($\lambda = 2, \alpha = 2$); (d) RTV [30] ($\lambda = 0.015, \sigma = 3$); (e) DSHFG [16] ($\lambda = 0.01$); (f) Ours ($\lambda_G = 0.02$)

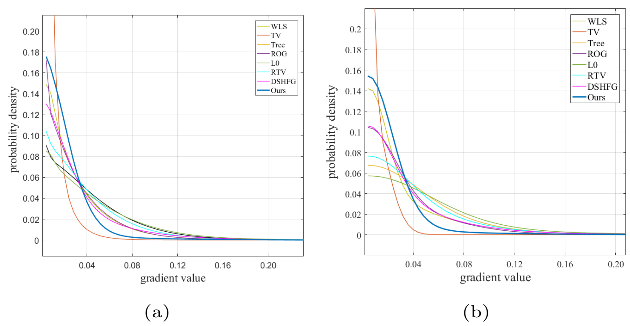


Fig. 18 Gradient value distribution: (a) average of all images; (b) distribution of 01_03.jpg

In summary, our algorithm outperforms the others in visual performance and is supported by the three suggested indexes: edge integrity rate, texture removal rate and gradient value distribution.

5 Application

5.1 Clip-Art Compression Artifact Removal

Image processing operations such as compression or super-resolution can distort images such as cartoons and clip-arts, and generate pseudo boundary that traditional denoising algorithms cannot remove. As can be seen in Fig.19 that image smoothing can effectively solve this problem and our method bings better results compared with others.

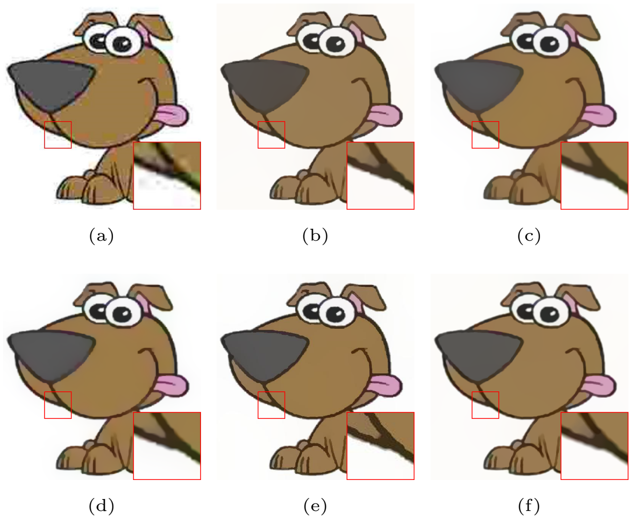


Fig. 19 Image abstraction. (a) original image; (b) L_0; (c) ROG; (d) RTV; (e) DSHFG; (f) Ours

5.2 Content-Aware Image Manipulation

Our proposed method can be combined with image significance detection [12] to realize content-aware image manipulation by dividing the image into foreground and background and processing them separately to achieve foreground enhancement or background blurring.

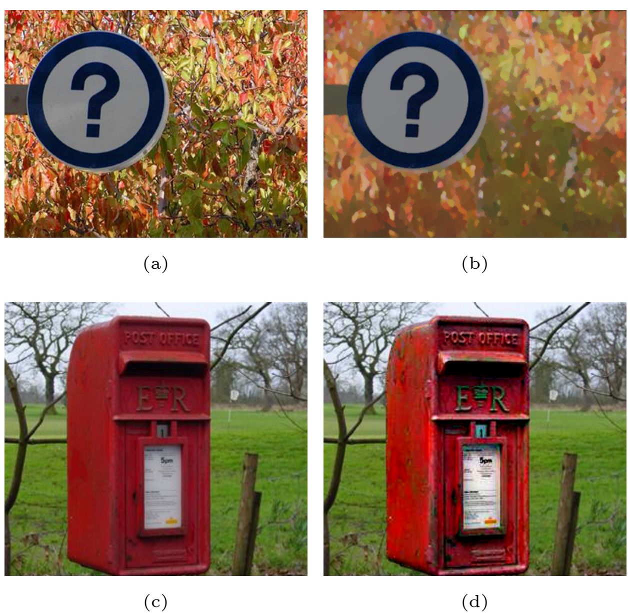


Fig. 20 Content-Aware Image Manipulation. (a) and (c): original images; (b): background blurring; (d): foreground enhancement

6 Conclusion and limitations

In summary, we make targeted improvements to three current problems in image smoothing: We enhance the smoothing performance in rich-textured images by pre-removal textures based on global sparse decomposition; By parameter adaptation based on patch-shift and parametric surface fitting through the improved Bessel, we solve the inconsistency of different parts of the image; Three evaluation indexes are proposed to evaluate smoothing performance quantitatively to get rid of the dependence on ground truth. The comparisons with the existing algorithms prove our algorithm works better. Besides, our algorithm also has limitations. We do not solve the problem of training pairs, so it cannot to train convolutional network intuitively. If this problem is solved, the smoothing quality can be further improved, which is what we will do next.

Acknowledgements

This work was supported by NSFC Joint Fund with Zhejiang Integration of Informatization and Industrialization under Key Project (U1609218).

References

[1] L. Bao, Y. Song, Q. Yang, H. Yuan, and G. Wang. Tree filtering: Efficient structure-preserving smoothing with a minimum spanning tree. *IEEE Transactions on Image Processing*, 23(2):555–569, 2013.

[2] S. Boyd, N. Parikh, E. Chu, B. Peleato, J. Eckstein, et al. Distributed optimization and statistical learning via the alternating direction method of multipliers. *Foundations and Trends® in Machine learning*, 3(1):1–122, 2011.

[3] B. Cai, X. Xing, and X. Xu. Edge/structure preserving smoothing via relativity-of-gaussian. In *2017 IEEE International Conference on Image Processing (ICIP)*, pages 250–254. IEEE, 2017.

[4] J. Chen, S. Paris, and F. Durand. Real-time edge-aware image processing with the bilateral grid. In *ACM Transactions on Graphics (TOG)*, volume 26, page 103. ACM, 2007.

[5] Q. Chen, J. Xu, and V. Koltun. Fast image processing with fully-convolutional networks. In *Proceedings of the*

IEEE International Conference on Computer Vision, pages 2497–2506, 2017.

[6] H. Cho, H. Lee, H. Kang, and S. Lee. Bilateral texture filtering. *ACM Transactions on Graphics (TOG)*, 33(4):128, 2014.

[7] J. Eckstein and D. P. Bertsekas. On the dougla-rachford splitting method and the proximal point algorithm for maximal monotone operators. *Mathematical Programming*, 55(1-3):293–318, 1992.

[8] Z. Farbman, R. Fattal, D. Lischinski, and R. Szeliski. Edge-preserving decompositions for multi-scale tone and detail manipulation. In *ACM Transactions on Graphics (TOG)*, volume 27, page 67. ACM, 2008.

[9] Z. Farbman, R. Fattal, D. Lischinski, and R. Szeliski. Edge-preserving decompositions for multi-scale tone and detail manipulation. In *ACM Transactions on Graphics (TOG)*, volume 27, page 67. ACM, 2008.

[10] S. Gu, Q. Xie, D. Meng, W. Zuo, X. Feng, and L. Zhang. Weighted nuclear norm minimization and its applications to low level vision. *International journal of computer vision*, 121(2):183–208, 2017.

[11] K. He, J. Sun, and X. Tang. Guided image filtering. *IEEE transactions on pattern analysis and machine intelligence*, 35(6):1397–1409, 2012.

[12] M. Jian, W. Zhang, H. Yu, C. Cui, X. Nie, H. Zhang, and Y. Yin. Saliency detection based on directional patches extraction and principal local color contrast. *Journal of Visual Communication and Image Representation*, 57:1–11, 2018.

[13] Y. Kim, B. Ham, M. N. Do, and K. Sohn. Structure-texture image decomposition using deep variational priors. *IEEE Transactions on Image Processing*, 28(6):2692–2704, 2018.

[14] T. Lindeberg. Scale-space theory: A basic tool for analyzing structures at different scales. *Journal of applied statistics*, 21(1-2):225–270, 1994.

[15] X. Liu, Y. Zhang, F. Bao, K. Shao, Z. Sun, and C. Zhang. Kernel-blending connection approximated by a neural network for image classification. *Computational Visual Media*, 6(4):467–476, 2020.

[16] G.-H. Ma, M.-L. Zhang, X.-M. Li, and C.-M. Zhang. Image smoothing based on image decomposition and sparse high frequency gradient. *Journal of Computer Science and Technology*, 33(3):502–510, 2018.

[17] K. Mikolajczyk and C. Schmid. An affine invariant interest point detector. In *European conference on computer vision*, pages 128–142. Springer, 2002.

[18] S. Ono. Edge-preserving filtering by projection onto l_0 gradient constraint. In *2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 1492–1496. IEEE, 2017.

[19] S. Ono. l_0 gradient projection. *IEEE Transactions on Image Processing*, 26(4):1554–1564, 2017.

[20] P. Perona and J. Malik. Scale-space and edge detection using anisotropic diffusion. *IEEE Transactions on pattern analysis and machine intelligence*, 12(7):629–639, 1990.

[21] L. I. Rudin, S. Osher, and E. Fatemi. Nonlinear total variation based noise removal algorithms. *Physica D: nonlinear phenomena*, 60(1-4):259–268, 1992.

[22] K. Subr, C. Soler, and F. Durand. Edge-preserving multiscale image decomposition based on local extrema. In *ACM Transactions on Graphics (TOG)*, volume 28, page 147. ACM, 2009.

[23] D. L. Sun and C. Fevotte. Alternating direction method of multipliers for non-negative matrix factorization with the beta-divergence. In *2014 IEEE international conference on acoustics, speech and signal processing (ICASSP)*, pages 6201–6205. IEEE, 2014.

[24] Y. Sun, S. Schaefer, and W. Wang. Image structure retrieval via l0 minimization. *IEEE Transactions on Visualization and Computer Graphics*, PP:1–1, 06 2017.

[25] C. Tomasi and R. Manduchi. Bilateral filtering for gray and color images. In *Iccv*, volume 98, page 2, 1998.

[26] Y. Wang, J. Yang, W. Yin, and Y. Zhang. A new alternating minimization algorithm for total variation image reconstruction. *SIAM Journal on Imaging Sciences*, 1(3):248–272, 2008.

[27] B. Weiss. Fast median and bilateral filtering. *Acm Transactions on Graphics (TOG)*, 25(3):519–526, 2006.

[28] L. Xu, C. Lu, Y. Xu, and J. Jia. Image smoothing via l_0 gradient minimization. In *ACM Transactions on Graphics (TOG)*, volume 30, page 174. ACM, 2011.

[29] L. Xu, J. Ren, Q. Yan, R. Liao, and J. Jia. Deep edge-aware filters. In *International Conference on Machine Learning*, pages 1669–1678, 2015.

[30] L. Xu, Q. Yan, Y. Xia, and J. Jia. Structure extraction from texture via relative total variation. *ACM Transactions on Graphics (TOG)*, 31(6):139, 2012.

[31] F. Zhang, J. Li, P. Liu, and H. Fan. Computing knots by quadratic and cubic polynomial curves. *Computational Visual Media*, 6(4):417–430, 2020.

[32] M. Zhang and C. Desrosiers. High-quality image restoration using low-rank patch regularization and global structure sparsity. *IEEE Transactions on Image Processing*, 28(2):868–879, 2018.

[33] R. Zhang and J. Kwok. Asynchronous distributed admm for consensus optimization. In *International Conference on Machine Learning*, pages 1701–1709, 2014.

[34] M. Zhao, H. Zhang, and J. Sun. A novel image retrieval method based on multi-trend structure descriptor. *Journal of Visual Communication and Image Representation*, 38:73–81, 2016.

[35] F. Zhu, Z. Liang, X. Jia, L. Zhang, and Y. Yu. A benchmark for edge-preserving image smoothing. *IEEE Transactions on Image Processing*, 28(7):3556–3570, 2019.

Xiang Ma is currently a Ph.D. student in the School of Software, Shandong University, Jinan. He received the Master's degrees from Shandong University, in 2020. His research interests include image processing and information visualization.

Xuemei Li received the Master's and Doctor's degrees from Shandong University, Jinan, China, in 2004 and 2010, respectively. She is currently an associate professor in the School of software, Shandong University, and a member of the GDIV lab. She is engaged in research on Geometric Modeling, CAGD, Image Processing and Information Visualization.

Yuanfeng Zhou received the masters and Ph.D. degrees from the School of Software, Shandong University, Jinan, China, in 2005 and 2009, respectively. He held a post-doctoral position with the Graphics Group, Department of Computer Science, the University of Hong Kong, Hong Kong, from 2009 to 2011. He is currently a Professor with the School of Software, Shandong University, where he is also a member of the GDIV Laboratory. His current research interests include computer graphics, information visualization, and image processing.

Caiming Zhang is a professor and doctoral supervisor of the Software College at Shandong University. He is now also the dean of the Digital media research institute at Shandong University of Finance and Economics. He received a BS and an ME in computer science from Shandong University in 1982 and 1984, respectively, and a Dr. Eng. degree in computer science from the Tokyo Institute of Technology, Japan, in 1994. From 1997 to 2000, Dr. Zhang had held visiting position at the University of Kentucky, USA. His research interests include CAGD, CG, information visualization and medical image processing.